Computational Methods for Comparative Analysis of Rare Cell Subsets in Flow Cytometry

by

Jacob Frelinger

Graduate Progam in Computational Biology and Bioinformatics
Duke University

Date: _____

Approved:

_____
Cliburn Chan, Supervisor

_____
Tom Kepler

_____
John Harer

_____
Kent Weinhold

_____
Mike West

Dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Graduate Program in Computational Biology and Bioinformatics
in the Graduate School of Duke University
2013

Abstract

Computational Methods for Comparative Analysis of Rare Cell Subsets in Flow Cytometry

by

Jacob Frelinger

Graduate Program in Computational Biology and Bioinformatics
Duke University

Date: _____

Approved:

_____
Cliburn Chan, Supervisor

_____
Tom Kepler

_____
John Harer

_____
Kent Weinhold

_____
Mike West

An abstract of a dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Graduate Program in Computational Biology and Bioinformatics
in the Graduate School of Duke University
2013

Copyright © 2013 by Jacob Frelinger
All rights reserved

Abstract

Automated analysis techniques for flow cytometry data can address many of the limitations of manual analysis by providing an objective approach for the identification of cellular subsets. While automated analysis has the potential to significantly improve automated analysis, challenges remain for automated methods in cross sample analysis for large scale studies. This thesis presents new methods for data normalization, sample enrichment for rare events of interest, and cell subset relabeling. These methods build upon and extend the use of Gaussian mixture models in automated flow cytometry analysis to enable practical large scale cell subset identification.

Contents

List of Tables

List of Figures

1

Introduction

Flow cytometry is an important tool in immune monitoring as it is the archetypal single cell assay for identifying cell populations. Flow cytometry works by individually labeling cells in a sample, typically by staining with fluorescent labeled monoclonal antibodies. These stained cells are then streamed at a rapid rate past a series of lasers and detectors. The lasers excite the fluorescent markers, and the resulting emission from the markers is recorded. These emission recordings indicate the presence and concentration of various cell surface and intracellular features. Using these fluorescence recordings it is possible to determine a wealth of information about the cells in the sample, including phenotype, activation state, and even specificity of cell surface receptors. The single cell resolution of flow cytometry allows it to work on large samples of heterogeneous cells, such as peripheral blood. This ability has made flow cytometry an integral tool in immune monitoring.

Currently most analysis of flow cytometry data is conducted via manual analysis through a process known as gating. In gating, populations of interest are separated visually from background populations by sequentially defining regions of interest in one or two dimensional projections in which the population of interest lies. Because of

the need to choose where to draw gate boundaries, as well as the order of the dimensions in which to gate the data, manual analysis can be highly subjective. Further, being limited to two dimensional projections can mean many gates are needed to identify cell populations that require multiple features to resolve. Large multi-center trials need to address the problems of how to train operators and standardize gating strategies. Because of these problems, much interest exists in developing automated analysis techniques that can objectively quantify cell subsets.

Automated analysis methods for flow cytometry can be broadly classified into those that use heuristics to identify cell subsets and those that use a statistical model of the data. Model based methods have proven to be an attractive method for automating identification of cellular subsets. In model based analysis, a generative probability model can be used to partition events into clusters. Many methods can be performed in the full dimensionality of the data set, allowing them to scale to an arbitrary number of dimensions. In addition, by using information from all dimensions simultaneously, these model based approaches can successfully identify some populations that are very difficult to separate using only two dimensional projections.

Significant effort has been expended in examining how to automate analysis of flow cytometry data, including the recent comparison of several automated analysis techniques in the FlowCAP challenge (Aghaeepour et al., 2013). A variety of methods to automate population identification exists in the literature. The GenePattern website provides a suite of online tools for working with and automating analysis of flow cytometry data (Spidlen et al., 2013). Sugr and Sealfon (2010) developed a unsupervised density contour clustering algorithm, called Misty Mountain. Zare et al. (2010) employed spectral clustering with a novel down sampling method to identify rare cells and small populations masked by larger populations. Artificial neural networks were used by Quinn et al. (2007) to examine the effects of erythropoietin on apoptosis and cell death in erythroid precursor cells in murine bone marrow.

Several methods based on K-means clustering have been developed. Some of the earliest work exploring clustering of flow cytometry data via K-means was by Murphy (1985). Aghaeepour et al. (2011) propose a modified K-means method that can identify concave cell populations by modeling a single population with multiple clusters. Zeng et al. (2007) used one dimensional histograms of the samples to help estimate the number of cluster components and guide multidimensional k-means clustering. A modified K-means method, called FLOCK, that automatically estimated the number of means and improved fit time was developed by Qian et al. (2010).

Several groups have used generative statistical models using mixture models. Mixture models are probability models comprised of a number of simpler distributions. Gaussian mixture models (GMMs) are a popular choice. Boedigheimer and Ferbas (2008) employed GMMs fit by expectation maximization to identify unique immunophenotypic features of B cell subsets in systemic lupus erythematosus patients. Chan et al. (2008) used GMMs fit by Markov chain Monte Carlo estimation to identify and cell subsets in human peripheral blood.

In addition to Gaussian mixture models, mixtures of other distributions have been employed. Lo et al. (2008) used a mixture of multivariate t distributions with a Box-Cox transformation fit by expectation maximization to identify the proportion of cells in various phases of the cell cycle and undergoing apoptosis. Finak et al. (2009) extended the work by Lo et al. (2008) with a method to merge components to better estimate the number of populations in a data set. Pyne et al. (2009) employed a mixture of skew t distributions estimated by expectation maximization to identify rare natural regulatory T cells in human peripheral blood.

For our work we have focused on mixtures of multi-variate normals, known as Gaussian mixture models. Given the set of means, covariances, and weights for each multivariate normal component in the Gaussian mixture model, the probability

density function is

$$p(x) = \sum_{i=1}^{N} \pi_i N(x|\mu_i, \Sigma_i) \tag{1.1}$$

where N is the number of component multivariate normals in the mixture model, μ_i and Σ_i are the mean and covariance of the i^{th} component normal distribution, and π_i is the weight of the i^{th} mixture component. Gaussian mixture models with enough component normals can approximate any distribution, including the complex multimodal distribution typically seen in flow cytometry data. Our work has included computational methods to speed up the estimation of the parameters of a Gaussian mixture model using massively parallel yet affordable graphics processing units (GPU) for speed-ups of two orders of magnitude, enabling the analysis of large data sets in a reasonable time span (Suchard et al., 2010).

While automated methods can significantly improve the accuracy and scalability of identifying cellular populations in flow cytometry data, challenges remain for the cross sample analysis necessary in large scale flow cytometry studies. Variability due to differences in cytometer setup in multi-center trials can lead to problems in comparing populations between samples. Very rare cell subsets can still be difficult to identify. Even once identified, cell subset labels are often arbitrary and not matched across samples. In this thesis, I present new methods to enable practical large scale cell subset identification that builds upon and enhances the use of Gaussian mixture models, including pre and post-processing steps to enhance rare cell subset identification via data normalization, biased sub-sampling, and cell subset relabeling.

Chapter 2 presents a data normalization method to reduce cross sample technical variability so as to facilitate direct cross sample comparison. Chapter 3 presents a biased sub-sampling method that leverages information in negative and positive controls to improve the detection of rare cell subsets. Chapter 4 presents a cluster relabeling method to facilitate direct comparisons of populations previously identified

by mixture models. In Chapter 5 a full pipeline is developed using these methods to illustrate the use of these methods in large scale flow cytometry analysis.

2

Normalizing Flow Cytometry Data

2.1 Introduction

The application of flow cytometry for immune monitoring or biomarker discovery in multi-center studies is hampered by the difficulty of harmonizing results of analysis performed in different laboratories. Because of differences in sample preparation and instrumentation, even "identical" panels may appear quite different. In particular, it can be hard to identify true differences due to biological variation across samples in the presence of variability due to various technical effects.

Some possible sources of technical variability arise from how samples are prepared, which reagents are used for analysis, and how the flow cytometer is calibrated. For example, differences in configuration of cytometer gain settings can have significant effects on fluorescent intensity as seen in Figure 2.1. To address the issue of cross-site technical variability, significant effort has been put into the standardization of flow cytometry assays by the research community. However despite all the effort put into standardization, this non-biological variation persists and flow cytometry data may be highly variable across sample batches, especially batches from multiple

laboratories.

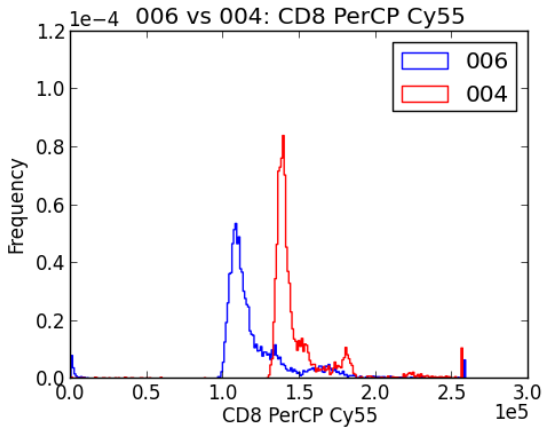

FIGURE 2.1: Differences in fluorescent intensity in 1x beads due to differences in voltage settings. Changing the voltage shifts and distorts the distribution of beads.

The reduction of variation due to technical effects would enable more direct cross-sample comparisons. Our goal with the methods presented in this chapter is to reduce these technical differences, leaving only differences attributable to the underlying biology. Minimizing the differences due to technical effects would enhance the reproducibility of flow cytometry data and make the use of flow cytometry more attractive and practical for use in multi-center clinical trials.

Experimentally, the most common procedure to control for between sample variation is the use of fluorescent beads with known emission properties. When these beads are run, the lasers are adjusted till the fluorescent values of the beads fall into specified target channels. While this is a good first step to control cross sample variation, setting of target channels on beads often does not fully align data when cell samples are run. There is also the possibility of "instrument drift" between bead calibration procedures. Beads are generally much brighter fluorescent objects than cells, and hence can be a poor example of the fluorescent intensities observed in cellular data.

We can also apply statistical procedures to normalize data. One simple method is to center the data, transforming the data from each lab to have a common mean and variance. This is achieved by transforming the observed values of data set X into X' by

$$X' = \frac{X - \bar{X}}{s_X} \tag{2.1}$$

where $\bar{X}$ is the sample mean of X and s_X is the sample variance of X.

Alternatively you can choose one data set X as a reference and set the mean and variance of a second data set Y to match X by transforming Y to Y'

$$Y' = \frac{Y - \bar{Y}}{s_Y} \cdot s_X + \bar{X} \tag{2.2}$$

where $\bar{X}$ and $\bar{Y}$ are the sample means of the X and Y data sets and s_X and s_Y are the sample variances of the X and Y data sets. The resulting Y' will have the same mean and variance as X. This has the advantage of keeping the data set close to the original scale of the data, facilitating visual interpretation of the data by flow cytometry experts.

While in many cases centering works well, it is sensitive to the common case where one or more cell populations are highly over-represented in one sample but not in the others. These over-represented populations can cause over spreading of the data when centering.

A common method in statistics is to transform the empirical distributions between data sets to be similar, using Quantile Normalization (Bolstad et al., 2003). However, the effects of quantile normalization can severely distort the data, which can be confusing for later visualization or manual analysis as seen in Figure 2.2

Alternative methods of normalization have been proposed. Hahne et al. (2010) proposed a method of feature alignment on a per channel basis. Features, typically chosen to be modes, or minima in the distribution of the data, are identified and

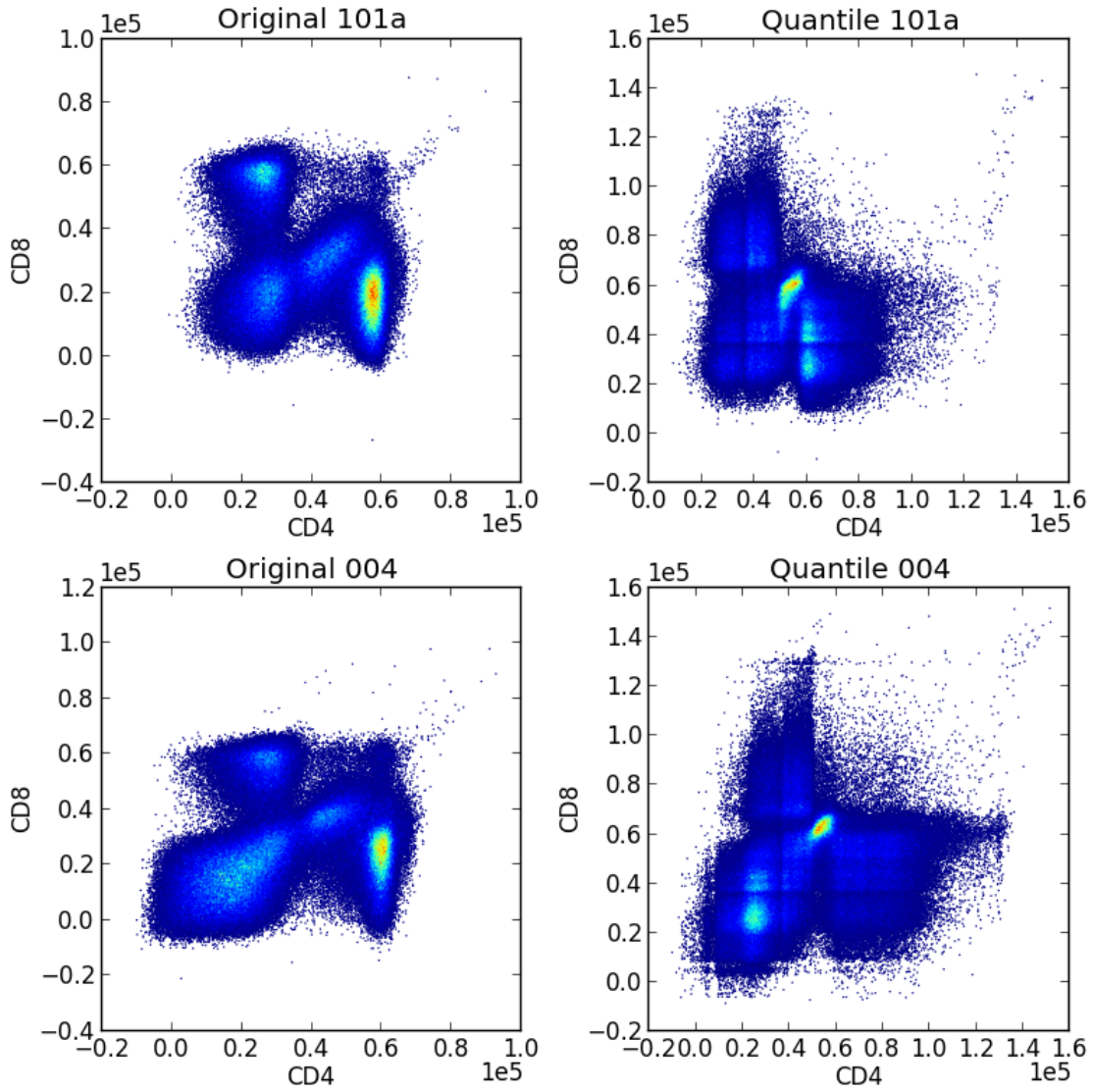

FIGURE 2.2: Quantile normalization between sample 004 and 101a from the EQAPOL data set. The left plots are the untransformed and the right column are the transformed data sets by quantile normalization. Quantile normalization significantly distorts the data which can cause problem for later visualization and manual analysis.

labeled across data sets. Events in the data sets are then warped such that features in the data sets then line up. While their method provides a method to align data sets, it is limited to working on only one channel at a time. It also requires estimating the

number of features in each channel and identification of the features in each channel, which can be challenging to automate.

All these methods perform univariate normalization and cannot account for effects that are correlated. In this chapter we propose a new alignment method that is multivariate, easily automated and minimizes distortion.

2.2 Approach

The goal of aligning two data sets is to adjust the distribution of the data so that each is "similar" to the other data set. We propose to minimize the difference between data sets using an affine transformation that can be used for data projections involving a linear transform and a translation operation. We use an affine transformation, as the properties of affine transformations are well understood.

To align two data sets, we assume that the differences observed between two samples can be approximated by an affine transformation. That is, we suppose the scatter and fluorescent observations in the first data set X are drawn from some distribution $f(\theta)$, and we assume that there exists a matrix A and vector b such that the second sample Y is drawn from $f(\theta) \cdot A + b$. In a multi-dimensional system of dimension k, A will be a matrix of size $k \times k$ and b will be a k dimensional vector.

If the matrix A is unconstrained, the transform can scale, reflect, rotate or shear the data, while the vector b controls translation of the data. The matrix A can be constrained to be a diagonal matrix of positive values, which will then only transform the scale of each dimension. This diagonal only alignment replicates individually aligning each dimension independently.

If we accept that the technical differences are described by an affine transformation, then the act of aligning the data sets is to find some A' and b' that reverses the affine transform. In other words , we need to find the A' and b' that minimize some dissimilarity measure between X and $Y \cdot A' + b'$. If we view the data sam-

ples as draws from an underlying probability distribution, it makes sense to use a measure of similarity between probability distributions such as the Kullback-Leibler divergence. Finding the A' and b' that minimizes the Kullback-Leibler divergence finds a transform that aligns the two data sets.

2.2.1 Kullback-Leibler divergence

Kullback-Leibler divergence, or D_{KL}, is a non-symmetric measure between two probability distributions, denoted P and Q, that describes the amount of information lost when Q is used as an approximation of P. For discrete distributions P and Q, the D_{KL} between P and Q is defined as

$$D_{KL}(P\|Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)} \tag{2.3}$$

and for continuous distributions P and Q the D_{KL} is defined as

$$D_{KL}(P\|Q) = \int_{-\infty}^{\infty} \log \frac{P(x)}{Q(x)} P(x)\,\mathrm{d}x \tag{2.4}$$

2.3 Algorithm

The basic algorithm for aligning data sets is performed in two steps. The first step is to estimate the underlying distribution X^* for data set X and Y^* for data set Y. The second step is to find an A and b such that the Kullback-Leibler divergence between X^* and $Y^* \cdot A + b$ is minimized. From this point, we will use A and b rather than A' and b' to reduce visual clutter.

2.3.1 Estimating Distributions

Estimating distributions using histograms

The simplest way to estimate the distribution that generates X is to use a K dimensional histogram of X^*. With appropriate bin sizes and a large number of events,

histograms produce reasonable estimates of the underlying distribution in the discrete space. Since the resulting distribution is discrete, it necessitates the use of the discrete Kullback-Leibler divergence, using each bin of the histogram as an element of the sample space.

While simple to calculate, the use of histograms to estimate the distribution has several drawbacks. Numerical optimizers make many estimates of A and b while trying to find the minimum Kullback-Leibler divergence. As each A and b is proposed, new histograms of $Y^* \cdot A + b$ need to be recalculated for each A and b along with calculating $Y^* \cdot A + b$, which in higher dimensional histograms can become computationally expensive.

In addition to the computational cost, there are stability issues with using histograms to estimate the distributions. The Kullback-Leibler divergence calculation involves calculation of the probability ratio between the two distributions. If the histogram representing the second distribution has a bin with no events while in the histogram representing the first distribution does, it will cause the Kullback-Leibler divergence to be undefined. One solution is to artificially add an event to every bin, similar to the Laplacian correction used in naive Bayes classifiers. Unfortunately, for high-dimensional systems with a large number of bins, this correction can induce significant bias in the results.

Estimating distributions using Gaussian mixture models

An alternative to using histograms for estimating the distributions of X and Y is to use mixture models. We have used a mixture of Gaussian distributions to describe our distributions. Given weights $\pi_1 \dots \pi_j$, means $\mu_1 \dots \mu_j$, and variances $\Sigma_1 \dots \Sigma_j$, we then describe the distribution as

$$p(x) = \sum_{i=1}^{j} \pi_i N(x; \mu_i, \Sigma_i). \tag{2.5}$$

One advantage of using a mixture of Gaussian distributions is that there exists a closed form for affine transformations of a mixture of Gaussians:

$$\left(\sum_{i=1}^{j} \pi_i N(x; \mu_i, \Sigma_i)\right) A + b = \sum_{i=1}^{j} \pi_i N(x; \mu_i A + b, A^T \Sigma_i A) \tag{2.6}$$

Applying transforms directly to the mixture model avoids having to re-estimate the distributions for each new A and b proposed. This allows us to work directly with distributions, and not the data sets described by those distributions. For very large data samples, this represents a significant performance improvement as opposed to to direct manipulation of raw data.

2.3.2 Kullback-Leibler divergence of Gaussian Mixtures

The D_{KL} between two k dimensional Gaussian distributions is easy to calculate as there exists a closed form for the D_{KL}

$$D_{KL}(P\|Q) = \frac{1}{2}\left(\mathrm{tr}\left(\Sigma_P^{-1}\Sigma_Q\right) + (\mu_Q - \mu_P)^\top \Sigma_Q^{-1}(\mu_Q - \mu_P) - k - \ln\left(\frac{\det \Sigma_P}{\det \Sigma_Q}\right)\right) \tag{2.7}$$

However, no closed form exists for the D_{KL} between mixtures of Gaussian distributions, and numerical approximation must be employed. It is possible to directly estimate the D_{KL} between mixtures by using Monte Carlo methods. We draw a large number of random samples from P and calculate

$$D_{MC}(P\|Q) = \frac{1}{n}\sum_{i=1}^{n} \log \frac{P(x_i)}{Q(x_i)} \tag{2.8}$$

where x_1 to x_n are the random draws from P. For large enough n, this will approximate the true Kullback-Leibler divergence.

However, the random noise introduced by using the Monte Carlo estimation of the D_{KL} often prevents numerical optimizers from converging. For optimization a

smooth approximation of the D_{KL} is needed. A suitable smooth function to approximate the D_{KL} is the variational lower bound described by Hershey and Olsen (2007) The lower bound of the D_{KL} between two mixtures of Gaussians is given by

$$D_V(P\|Q) = \sum_j \tau_j \log \frac{\sum_k \tau_k e^{-D_{KL}(p_j\|p_k)}}{\sum_k \pi_k e^{-D_{KL}(p_j\|q_k)}} \tag{2.9}$$

where p_j and q_k are the individual Gaussian mixture components of P, and Q τ_z is the weight of component z in P, π_z is the weight of component z in Q , and $D_{KL}(p_j\|q_k)$ is the Kullback-Leibler divergence between component clusters.

2.3.3 Gradient

Many optimizers make use of the gradient of the objective function to perform optimization. This gradient can be estimated via numerical methods automatically, or provided in closed form. Providing an analytical form of the gradient can greatly improve the speed of the optimizers (Table 2.1 and 2.2) and has the potential to increase the accuracy of the estimated optima. Differentiating Equation 2.9, the gradient of the lower bound of the Kullback-Leibler divergence is given by

$$\frac{\partial D_V(P\|Q)}{\partial \theta} = \sum_j \tau_j \frac{\sum_k \pi_k e^{-D_{KL}(p_j\|q_k)} \frac{\partial D_{KL}(p_j\|q_k)}{\partial \theta}}{\sum_k \pi_k e^{-D_{KL}(p_j\|q_k)}} \tag{2.10}$$

where p_j and q_k are the individual Gaussian mixture components of P and Q, τ_z is the weight of component z in P, π_z is the weight of component z in Q, and $D_{KL}(p_j\|q_k)$ is the Kullback-Leibler divergence between component clusters. The derivative of the Kullback-Leibler divergence between the two mixture components $p_j = N(\xi_j, V_j)$ and $q_k = N(\mu_k A + b, A^T \Sigma_k A)$ is given by

$$\begin{aligned}\frac{\partial D_{KL}(p_j\|q_k)}{\partial A} = &-\Big(-\Sigma_k A + \mu_k^T(\xi_j - \mu_k A - b) + \Sigma_k A (A^T\Sigma_k A)^{-1} \times \\ &\big[V_j + (\xi_j - \mu_k A - b)^T(\xi_j - \mu_k A - b)\big]\Big)(A^T\Sigma_k A)^{-1}\end{aligned} \tag{2.11}$$

$$\frac{\partial D_{KL}(p_j\|q_k)}{\partial b} = -(\xi_j - \mu_k A - b)(A^T\Sigma_k A)^{-1} \tag{2.12}$$

Table 2.1: Speed of alignment in seconds of a three dimensional synthetic data set with 2 to 10 clusters. First column indicates the number of clusters being aligned. Second column is the final value of the estimated Kullback-Leibler divergence using the analytic gradient. Third column is the time taken using the analytic gradient. Fourth column is the final value of the estimated Kullback-Leibler divergence using numerical approximations to the gradient. The final column is the time taken in seconds to align the two samples using numerical approximations of the gradient. Using the analytic gradient improves the speed of aligning two samples over estimating the gradient.

	Analytic Gradient		Estimated Gradient	
Number of clusters	Obj.	Time	Obj.	Time
2	0.0000	1.0771	0.0000	1.9008
3	0.0000	1.9201	0.0000	5.3483
4	0.0000	4.4766	0.0000	9.1116
5	0.8866	7.0122	0.8866	13.6436
6	0.9166	8.9722	0.9166	19.5285
7	0.0000	13.1239	0.0000	24.4226
8	0.8940	17.9498	0.9929	34.1058
9	1.0318	19.6191	1.0318	44.5129
10	1.0770	21.6603	1.0770	41.8715

2.4 Results

Normalization by affine mapping was performed on both synthetic and real data sets. The Python OpenOpt framework, using the 'ralg' optimizer based on the r-algorithm method developed by Shor and Zhurbenko (1971), was used as it proved robust and had reasonable computational speed. Other optimizers were tested and found to either have higher final objective functions or were slower to converge.

2.4.1 Synthetic data

Alignment of synthetic one dimensional samples with missing populations

To illustrate alignment we start with a simple example using one dimensional data shown in Figure 2.3. The samples represent the common case where a population exists in one data set but is missing in the other, such as when comparing one sample against a negative control. For the reference sample draw events from one of two

Table 2.2: Speed of alignment in seconds of a five component data synthetic data sets based on the number of dimensions ranging from 2 to 10. First column indicates the dimension of the data sets being aligned. Second column is the final value of the estimated Kullback-Leibler divergence using the analytic gradient. Third column is the time taken in seconds using the analytic gradient. Fourth column is the final value of the estimated Kullback-Leibler divergence using numerical approximations to the gradient. The final column is the time taken in seconds to align the two samples using numerical approximations of the gradient. Using the analytic gradient significantly improves the speed over estimating the gradient as the number of dimensions increases

	With Gradient		Without Gradient	
Dimension	Obj.	Time	Obj.	Time
2	0.6661	4.4281	0.6661	7.4290
3	0.8310	6.2651	0.8310	13.2366
4	0.8310	7.0143	0.8310	19.5441
5	0.8310	8.3618	0.8310	22.5654
6	0.0000	6.8011	0.0000	23.5626
7	0.8310	8.0450	0.8310	36.3744
8	0.0000	11.2076	0.0000	31.3104
9	0.0000	10.4891	0.0000	37.6305
10	0.0000	11.1432	0.0000	51.5135

normal distributions. Ten thousand events are drawn from a normal distribution with mean 2 and a variance of 1 to simulate a negative cell population, and 300 events are drawn from a normal distribution with mean 10 and variance 0.5 to simulate a positive cell population. The target sample consists of 10,000 events drawn from a normal distribution with mean 4 and variance 0.5. Because of the small "positive" population in the reference sample increasing the total population variance, centering causes "over-spreading" of the target population. Alignment using an estimated A of 2.36441 and b of -7.22463 avoids this over spreading. Outlier populations such as the small 300 event population that only exist in one data set are not uncommon in flow cytometry, where activation of a specific cell type causes new populations to form that do not exist in control or non-reactive samples. These populations are often of interest, as they represent biological differences between samples.

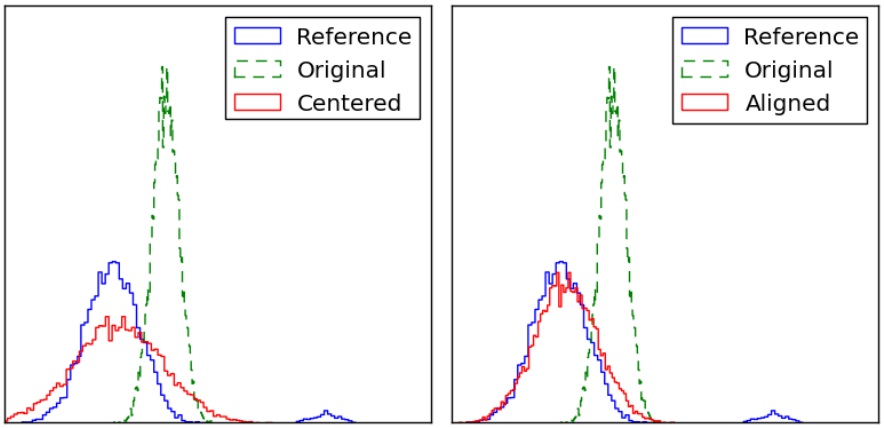

FIGURE 2.3: Comparison of centering (re-scaling so that the data sets have the same mean and variance) to alignment for for two synthetic data sets. The reference data set consists of 10,000 events drawn from a normally distributed around mean 2 with a variance of 1 and 300 events normally distributed around mean 10 with variance 0.5. The data set being aligned consists of 10,000 events normally distributed around mean 4 with variance 0.5. A and b were estimated to be 2.36441 and -7.22463 respectively.

Alignment of two dimensional synthetic data

In Figure 2.4, a synthetic two dimensional data set was created comprising two Gaussians, shown in the top left plot as X. It was then scaled, sheared and translated into data set Y. Alignment constrained to the diagonal entries, and full rank alignment was performed. While constrained diagonal alignment can successfully scale the data to the correct locations, the shape of the individual Gaussian components is distorted from the original data. Full rank alignment successfully scales and shears the data to not only have the correct location, but also the correct shape.

While full rank alignment can provide better results, it is significantly more computationally expensive. In the two dimensional case, the diagonal constrained alignment only needs to estimate four parameters, while full rank must estimate six. In a d dimensional case, diagonal alignment needs to estimate $2d$ parameters, while full alignment must estimate $d^2 + d$. This trade-off needs to be considered when choosing

between full rank alignment and diagonal alignment.

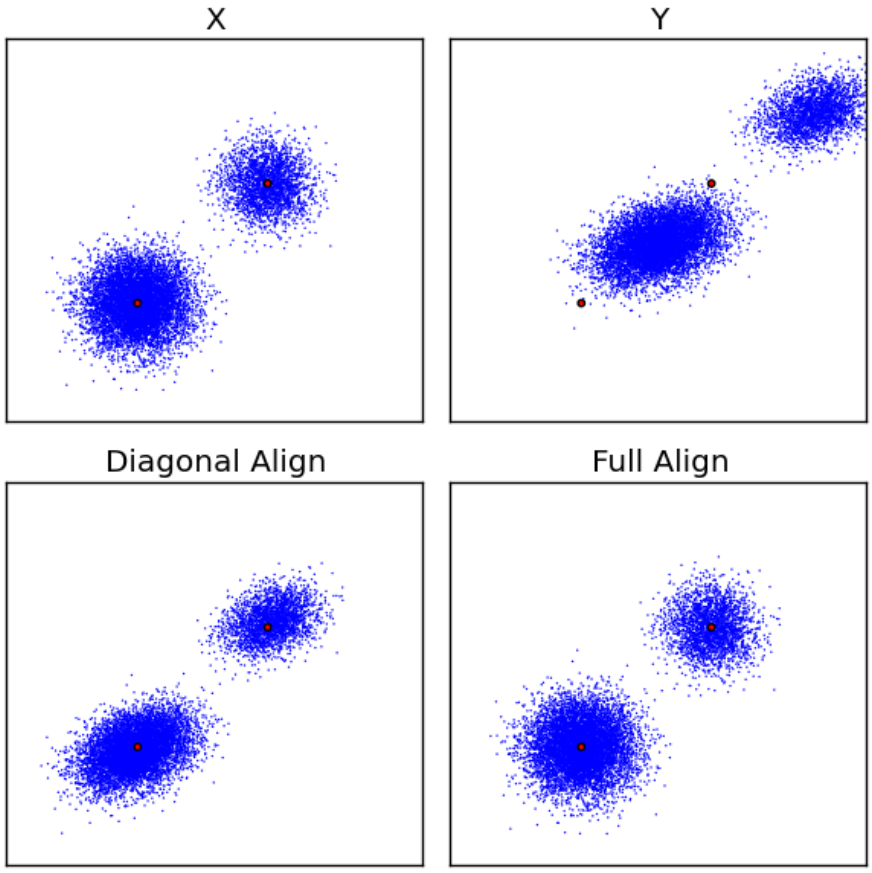

FIGURE 2.4: Comparison of diagonal and full alignment for a synthetic two dimensional data set. The top left plot shows the reference data X, comprising two Gaussians. The top right plot shows Y, a scaled, and sheared transformed X. The bottom left panel shows the results of diagonally aligning Y to X. The bottom right plot shows the full rank alignment of Y to X. Red dots are the means of the original Gaussian mixture components in X.

2.4.2 Experimental data

We next applied the affine normalization method to data from the External Quality Assurance Program Oversight Laboratory (EQAPOL) flow cytometry proficiency evaluation. The EQAPOL send outs are part of a study of the evaluation of the performance of multiple laboratories. The samples used are from an intracellular

cytokine staining (ICS) assay sent to multiple laboratories, where each laboratory received identical samples and lyophilized reagents to run with standardized flow cytometry panels. In other words, exactly the same subject, sample, reagents and protocol are used by all laboratories. Differences observed between matched samples should primarily be due to technical differences.

The EQAPOL data sets consist of 27 samples from 12 labs. The 27 samples from each lab consist of 9 samples from three subjects. The three subjects were common across all labs. Each of the samples from each subject included two stimulation samples and a brefeldin only negative control sample. Each of these samples has two technical replicates. A total of 324 samples were processed. Subjects were labeled A,C, and E. Samples number 1,2 and 3 are the brefeldin only negative control and technical replicates. Samples numbered 5,6,7 are a cytomegalovirus peptide pool stimulation (CMV pp65) sample and technical replicates. Samples numbered 9,10,11 are a cytomegalovirus, Epstein-Barr virus and influenza virus (CEF) epitopic peptide stimulation and technical replicates.

Affine normalization was performed to compare aligning two matched samples in one dimension, and in two dimensions. Affine normalization was also used to align 15 matched samples across multiple laboratories in six dimensions. Finally, affine normalization was used to align two matched samples and the affine transform generated applied to the other samples from the lab.

One dimensional alignment of CD4 channel

In the one dimensional example we normalized the CD4 FITC channel and compared the A01 brefeldin only negative control from laboratories 001 and 003. The sample from laboratory 003 was used as a reference, and the affine normalization of the sample from laboratory 001 was estimated. Results of centering and affine normalization are shown in Figure 2.5. The centering transform does not line the

CD4 positive population up well, nor align the modes of the negative populations, due to the long tail on the left of the negative population in the reference sample. In contrast, affine alignment using a three component mixture model shows good overlap in the positive populations and brings the negative modes into closer alignment than centering does.

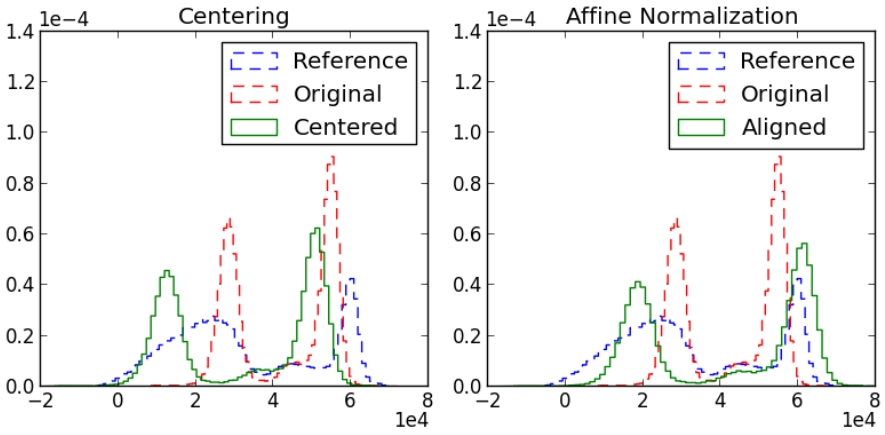

FIGURE 2.5: Comparison of centering (re-scaling so that the data sets have the same mean and variance) to affine normalization for sample 003_A01 (Target) and 001_A01 from the EQAPOL data send out. A and b were estimated to be 1.60736 and -27028.3016 respectively.

Alignment of Two fluorescent channels

For the case of two dimensional alignment, the A01 brefeldin only negative control sample from laboratory 001 was aligned to the matching sample from laboratory 003, using the CD3 APC and CD8 PerCP Cy55 channels (see Figure 2.6). Compensation was performed individually on each sample to remove spillover prior to affine normalization. Before affine normalization or centering, the CD3 negative population from laboratory 001 is a little higher in both CD3 and CD8, while the CD3 positive population is a little lower in CD3 than in the sample from laboratory 003. Of particular note is the hyper negative population in the sample from laboratory 003

indicated by the arrow. This population poses a problem for centering, as it greatly pulls the 001_A01 centered population too far down in CD3 intensity.

With the optimized parameters values of $A' = \begin{bmatrix} 1.51328784 & 0.01311331 \\ 0.12438734 & 1.10318245 \end{bmatrix}$ and $b' = \begin{bmatrix} -28238.65004973 & -5970.42079198 \end{bmatrix}$, the CD3 negative population in 001_A01 aligned with the negative population in 003_A01, and similarly the CD8 positive CD3 positive population.

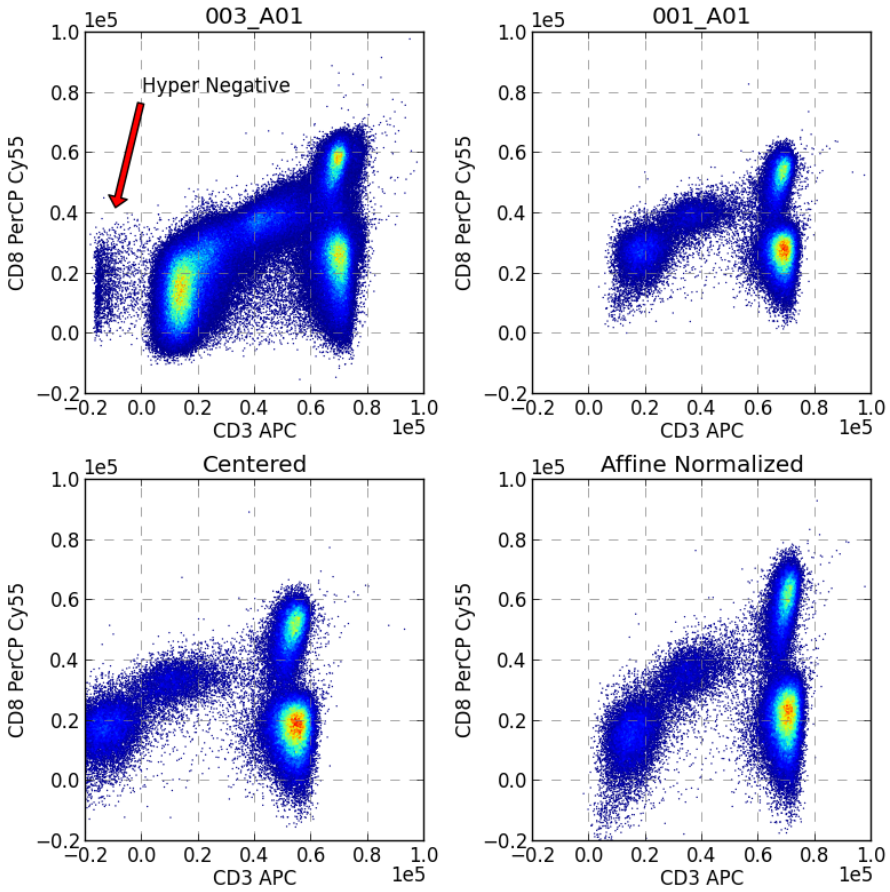

FIGURE 2.6: Comparison of centering and aligning including shearing. The red arrow in the first plot shows the position of a hyper negative population in sample 003_A01. The reference sample is 003_A01 and the aligned sample is 001_A01 from EQAPOL data send-out. Panel 3 shows how the lack of mass in the CD3 negative and hyper negative population from 001_A01 compared to 003_001 causes it to be pull too far to the left when centering. Alignment effectively transforms the data set so that they overlap each effectively.

Aligning data across 15 laboratories

The affine normalization algorithm proposed can be scaled to fit multiple data sets by choosing a reference data set and aligning the other data sets to match the reference data set, as seen in Figure 2.7. Here we chose laboratory 101 to be the reference lab and align fourteen brefeldin only negative control samples from other laboratories to it. Figure 2.8 shows a detailed view of how the alignment of data set from laboratory 008 is improved as the modes in each of populations fall closer in alignment with the modes in the reference data set. For this alignment the submatrix of A corresponding to the scatter channels was constrained to only have diagonal element, as shear and rotational effects are not believed to happen between the fluorescent and scatter channels. The fluorescent channels were not constrained, and both diagonals and off diagonals of the A matrix for these channels were estimated.

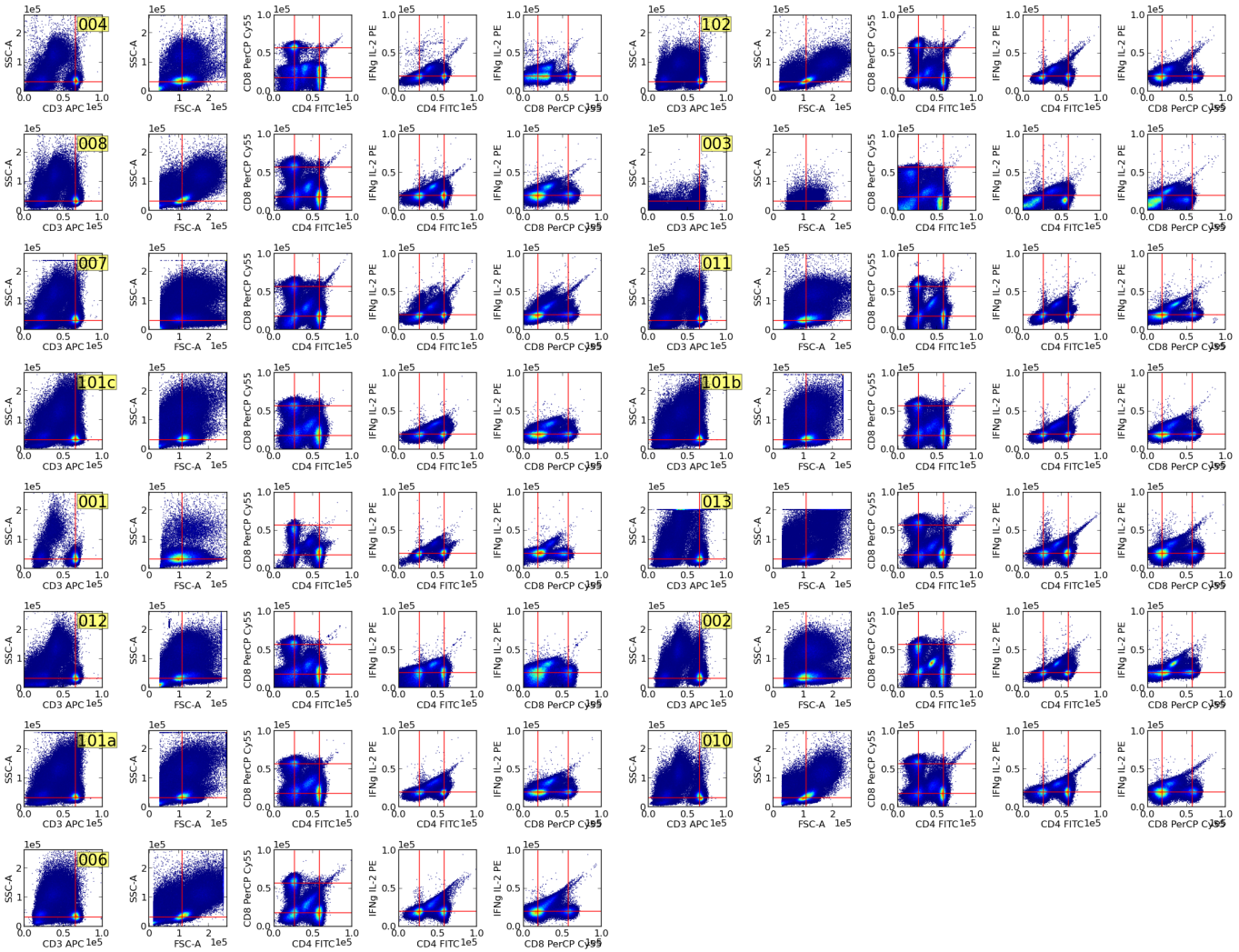

FIGURE 2.7: Alignment of the 15 files in the EQAPOL data set. Files were aligned to the sample labeled 101a. Red lines indicate the modes of populations in the 101a data set. The A matrix was constrained to be 0 for the off diagonals for the two scatter channels, FSC and SSC, while off diagonal entries were estimated for the remaining fluorescent channels.

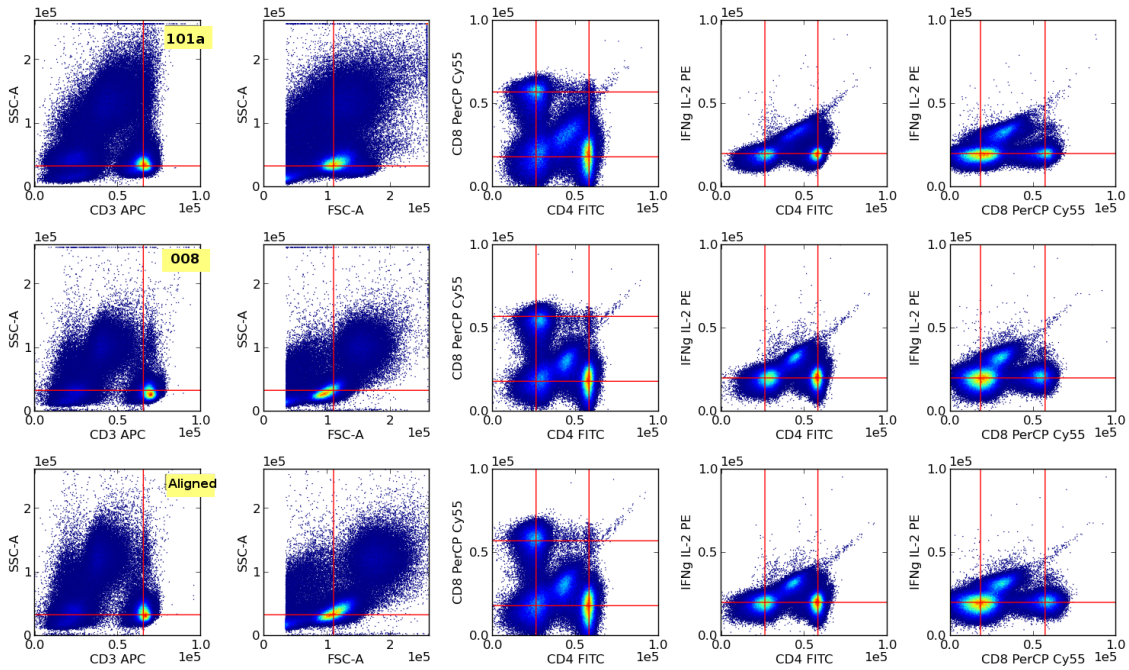

FIGURE 2.8: Alignment of data sample 008 to 101a from the EQAPOL data set. The top row contains the target data set 101a. The second row is the unaligned events from sample 008, and the bottom row shows the data set after alignment. Red lines indicate the approximate location of modes of populations in the 101a data set. The A matrix was constrained to be 0 for the off diagonals for the two scatter channels, FSC and SSC, while off diagonal entries were estimated for the remaining fluorescent channels.

Using a common transform within 12 laboratories

The affine normalization algorithm was used to align all samples in the EQAPOL proficiency data set. The send out comprises 12 sites with 27 samples per site. To avoid the computational time of aligning over 300 samples, only a single sample from each of the 12 sites was aligned to a common reference. The transform generated for each site was then used to transform the remaining samples for that site. Using a common transform for each site assumes that the technical error is constant within a site. An example data set, sample E05 from lab 010 is presented in Figure 2.9. Alignment was performed between the E06 samples from each lab, and then applied

to the other samples from the lab. By assuming minimal technical differences between samples from within a laboratory, only twelve alignments need to be run instead of the full 324, saving a significant amount of computational time.

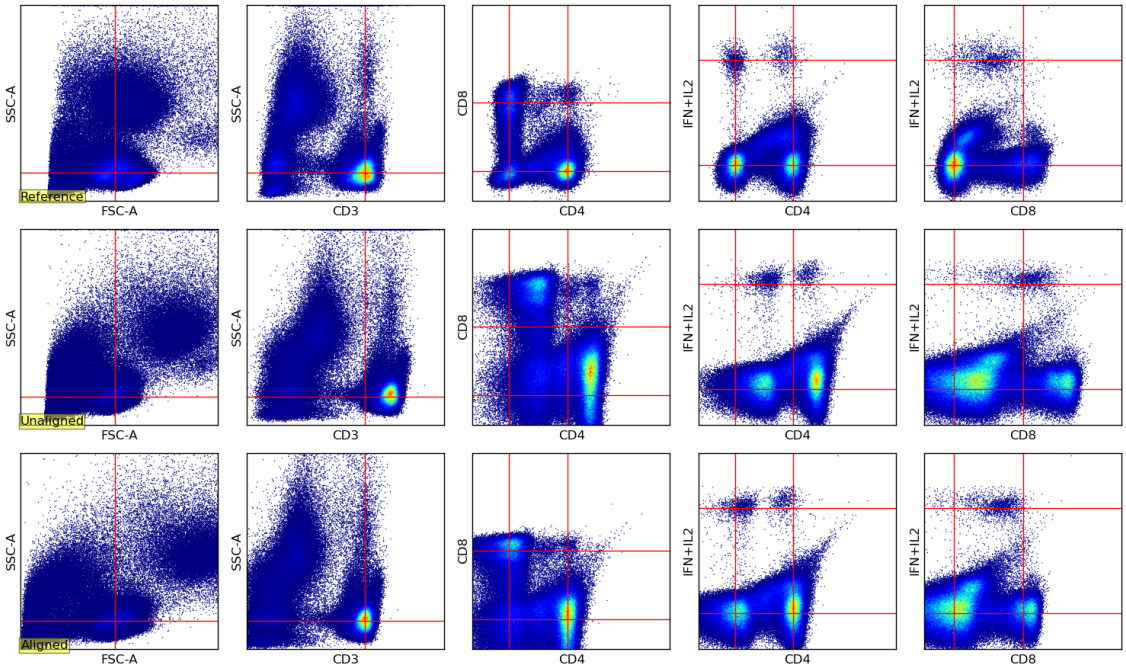

FIGURE 2.9: Alignment of samples from lab 010 to lab 101 using a common transform. Sample E06 from lab 010 was aligned to the corresponding sample E06 from lab 101. The resulting A and b were then applied to the other samples from lab 010. Displayed in the top row is the E06 sample from lab 101 used as the reference sample for alignment. The middle row shows an unaligned sample E05 from lab 010. The bottom row shows the transformed E05 from lab 010 using the transform generated from aligning samples E06 from lab 101 and lab 010. Red lines highlight the position of features in the reference data set.

2.5 Discussion

The goal of much flow cytometry analysis is the comparison of cell subset relative frequencies across different data samples, from simple treatment versus control samples, to longitudinal studies with many time points. Large scale multi-center clinical studies can involve hundreds of participants with many samples from each subject

run at different institutions or labs. This inevitably leads to difficulties comparing across samples due to differences between cytometers at different locations, and differences due to samples being run on different days. Aligning data sets to each other attempts to remove this variation and facilitate cross sample analysis.

The proposed alignment method requires the nomination of a data set to be the reference. The resulting aligned data sets will try and match the distribution of the reference data set. A poor choice of reference data set, one that has significant problems being analyzed or that does not have a similar distribution to all the other data sets, will cause difficulty in analyzing data sets aligned to it. To choose a reference sample we propose calculating the sum of Kullback-Leibler divergences between each proposed reference sample and the other samples. The sample with the minimal sum of Kullback-Leibler divergences is then the "most similar" to all the other samples and a prime candidate for being used as a reference.

Numerical optimizers used are sensitive to initial conditions, and can find non optimal solutions due to getting stuck in local minima or maxima. To alleviate this, careful choice of initial conditions is needed, and we have defaulted to centering the data sets following Equation 2.1 such that they have a common global mean and variance before affine normalization is performed. Using multiple random starting conditions can also help determine the optimal alignment between data sets.

In addition to getting stuck in local minima, the optimizer is also sensitive to the ability of the estimated mixture models to accurately describe the data. Significant under-fitted mixture models poorly describe the data set and can lead to inaccurate alignment. This must be balanced with the computational cost of getting better model fits and issues with underruns in the calculation of the probability density caused by small very components.

Finally, the proposed alignment algorithm assumes some affine transform can approximate the between sample error well. Hence it cannot account for non-linear

error. Assuming a non-linear error form greatly increases the complexity of the problem of alignment.

On the surface, alignment should also be able to remove the need to compensate multiple samples. Only one sample would need to be compensated, and the remaining samples could be aligned to the compensated sample. However, because of how the mixture models are fitted on transformed data, this will not work as currently described. Flow cytometry data is first transformed via log, or log-like transforms. This transform spreads out data making it easier to model cell populations. The model fitting routines employed for alignment work well in this transformed space. Because compensation is performed before the log or log-like transform is applied, the affine normalization routine cannot estimate the original compensation after the non-linear log or log-like transform. It might be possible to project the estimated mixture models used for alignment fit in the transformed space back into the untransformed space, making them a mixture of log-normal distributions. This would also require methods to calculate the Kullback-Leibler divergence between mixtures of log-normals and possibly the gradient of the divergence function. Alternatively models could be developed that will fit the highly skewed untransformed data well, and the affine normalization method applied to those models.

3

Exploiting Biological Controls to Enrich for Rare Events in Flow Cytometry Data

3.1 Introduction

Many of the cell subsets of interest in flow cytometry are found in frequencies well below 1% of all the cells in a sample. To reliably detect cells that are this infrequent, large numbers of cells need to be collected. In addition to the low frequency of these cells of interest, multiple fluorescent markers are often necessary to differentiate these rare populations from other populations. Unfortunately, the volume and high dimensionality of data needed to be collected to detect such rare subsets can result in very long run-times for statistical model fitting.

Furthermore, these rare cell subsets can be masked by larger nearby populations, making detection of rare cell subsets difficult. The proximity of these rare cells to larger background populations makes detection difficult, as the rare population often lies in the tail of the distribution of the larger background population. The tail of the background distribution then masks the smaller rare cell population. Masking is particularly problematic for heavy tailed mixtures such as mixtures of Student's t

distributions, as the small masked populations are easily absorbed in the larger tails. If the density of the rare population could be increased relative to the background population, it would then stand out more and be easier to detect.

While mixture modeling approaches address many issues in analysis of flow cytometry data, the run-times of such models typically scale linearly with the number of events, and quadratically with the number of markers. To describe the complex multi-modal data distributions typically seen in polychromatic flow cytometry data, mixture models need a large number of components. As a result, the run-time for estimating mixture models can be a bottleneck for analysis. Subsampling the data will reduce the run-time by reducing the number of events in the sample, but care must be taken when subsampling not to lose populations of interest or grossly distort the original distribution of the data.

3.1.1 Subsampling methods

Subsampling reduces the data size, but overly aggressive subsampling will lose rare populations of interest. This trade off must be carefully balanced when reducing the data set. By biasing the subsampling to prefer events of interest, more aggressive subsampling can be performed while preserving the features of interest. However, all biased subsampling methods can grossly distort the distribution of the data and lead to inappropriate results.

Uniform subsampling

The simplest form of random subsampling is uniform subsampling, in which each event in the original data set has an equal probability of ending up in the subsample. As shown in Figure 3.1, this preserves the relative frequency of cellular populations and the distribution of those populations. However, nothing is done to preserve rare events, and hence rare populations run the risk of not being represented if the sample

size is too small.

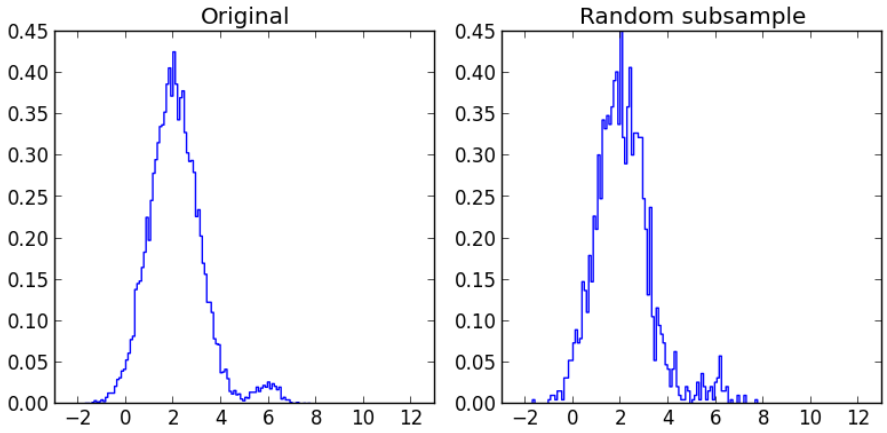

FIGURE 3.1: Effects of random subsampling on the distribution of data. The left plot shows a normalized histogram of 10300 events split between two clusters, with 10000 events in the left cluster and 300 events in the right cluster.. The right plot shows the normalized histogram of 2000 random drawn events from the original data. The relative frequency of the smaller cluster is unchanged by subsampling

Density based subsampling

An alternative to uniform subsampling is to bias the sample based on density information in the data set. In density based subsampling the probability of being drawn is a function of the local density. To enrich for rare events, we can select events without replacement from the data set with probability reciprocally proportional to the local density,

$$p(X) \propto \frac{1}{q(X)} \tag{3.1}$$

where $q(X)$ is the estimated posterior probability. Once the density for each event is estimated, the reciprocal densities are normalized to sum to one, and used as the weights to draw events without replacement. By biasing the sample in favor of events in low density areas, the hope is to preserve rare events, while reducing the frequency of other events, as seen in Figure 3.2. This results in a sample that contains

a majority of the rare events of interest and a low number of total events to improve computational run-times. A downside to density based biased subsampling is that any rare event, including those that are not desired, such as debris or unmasked rare frequency events in regions that are not of interest, are also preserved in the sample. These unwanted rare events potentially increase the rate of false positives. In particular, events on the boundary of large distributions could lead to flattening of the distributions.

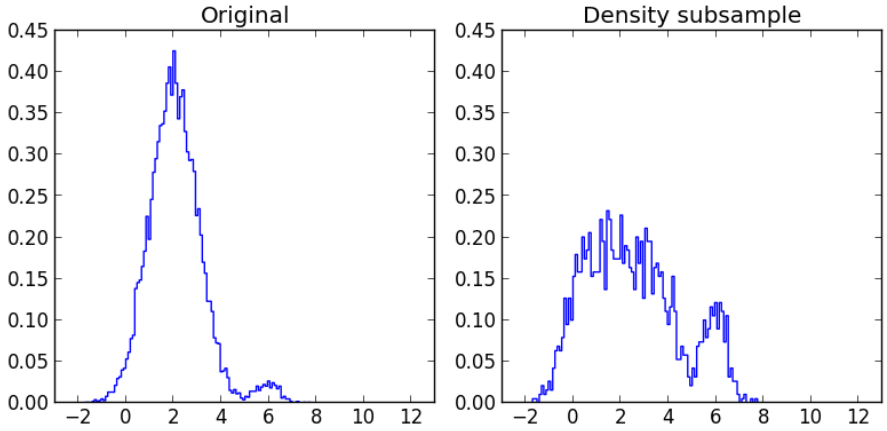

FIGURE 3.2: Effects of density based subsampling on the distribution of data. The left plot shows a normalized histogram of 10300 events split between two clusters. The right plot shows the normalized histogram of 2000 random drawn events inversely proportional to their estimated density in the original data. The relative frequency of the smaller cluster is increased, but the distribution of the larger population is distorted.

SPADE subsampling

The SPADE package, described by Qiu et al. (2011), contains a method of density based subsampling data as one step in the processes of analyzing flow cytometry data. Subsampling in SPADE works by maintaining a window of acceptable local density over all events. If the local density around an event is too low, it is removed as an outlier. If the local density around an event is too high, the probability of it being included in the subsample is the ratio of the upper bound of acceptable density

and the current density (see Equation 3.2). Events that have a local density between the lower bound outlier density and an upper bound target density are automatically included. They are, therefore, subsampled using the formula

$$p(\text{keeping } X|LD_X) = \begin{cases} 0 & \text{if } LD_X \leqslant OD \\ 1 & \text{if } OD < LD_X \leqslant TD \\ \frac{TD}{LD_X} & \text{if } TD < LD_X \end{cases} \tag{3.2}$$

where LD_X is the local density for event X, OD is the minimum density to not be considered an outlier, and TD is the upper bound of acceptable local density. Similar to density subsampling, the subsampling routine in SPADE increases the frequency of rare events, but significantly distorts the frequency of the larger population as shown in Figure 3.3. Another weakness is the need to specify the outlier density, OD, and target density,TD.

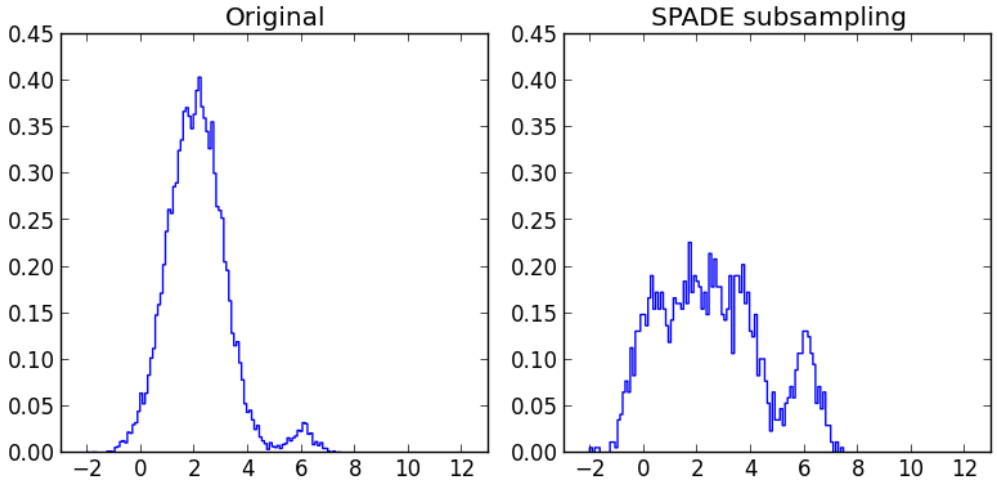

FIGURE 3.3: The subsampling step of SPADE's effect on the distribution of events. The left plot shows a normalized histogram of 10300 events split between two clusters with 10000 events coming from the left cluster and 300 from the right. The right plot shows the normalized histogram of 1768 random drawn events from the original sample using the method used in SPADE with a target density of 2 and an outlier density of 1. The relative frequency of the smaller cluster is increased, but the distribution of the larger population is distorted.

3.2 Approach

As we have seen, unbiased subsampling has a trade off between sample size and detection of rare events. Too few events in the reduced sample and rare populations are no longer represented; too many and minimal speed up in analysis is realized. Biased subsampling attempts to mitigate this trade-off by ensuring events of interest are still well represented while reducing the overall number of events in the sample. However, density based subsampling induces a bias that can distort the estimated cell subset frequencies of many populations in analysis.

To overcome problems with density based subsampling, we propose to exploit information from positive and negative controls to help identify the cells of interest. Well designed flow cytometry experiments usually have a negative control, and many include a positive control as well. Negative controls are used define a baseline. The events of interest are, by definition, either not found or extremely rarely found in the negative control. Therefore, subsampling from *anomalous* events, defined as the events in the data set whose probability of coming from the negative control is low, will enrich the data set for the rare cell subsets we are interested in. One downside is that not all anomalous events are likely to be of biological interest, as rare events in the tails of distributions of all populations will be seen as anomalous events.

To further refine detection of rare events, we can additionally incorporate information in positive controls into our anomalous event determination to define regions of *interesting* events. In the context of flow cytometry, positive controls are designed such that events from all cell subsets of biological interest are found in relatively high frequency. Events found in regions where events exist in the positive control and do not exist in negative controls, or where the frequency of cells differs between the negative and positive control are regions of interest, and typically include the desired rare cell subsets. By biasing the subsampling to over-represent these inter-

esting events, we can preserve the very rare cell subsets, while limiting total sample size to speed up computation.

3.3 Algorithm

3.3.1 Anomaly subsampling

In anomaly subsampling, events from the data set are selected inversely proportional to the posterior probability of being from the negative control

$$p(X) \propto \frac{1}{q(X|\theta_{Negative})} \tag{3.3}$$

where $q(X|\theta_{Negative})$ is the probability of X coming from the negative control. A Gaussian mixture model is fitted to the negative control, and the probability of each event in the data set is calculated. The reciprocal of the probabilities are normalized to sum to one, and used as the weights to randomly draw events without replacement. The effect of using reciprocal probabilities causes events with low probability of coming from the negative control distribution to be highly likely to be included in the sample as seen in Figure 3.4. We see that anomaly subsampling distorts the distribution similarly to density based subsampling.

3.3.2 Interesting event subsampling

To identify regions of interest, first mixture models are fitted to both positive and negative controls. The log-posterior probability of each event in the data is calculated for the estimated distributions of the negative and positive controls. The log posterior probability ratio of the event coming from the positive control versus the negative control is then normalized and used as a weight to draw samples from the original data set without replacement. Therefore, the probability of being sampled is proportional to the log posterior probability ratio

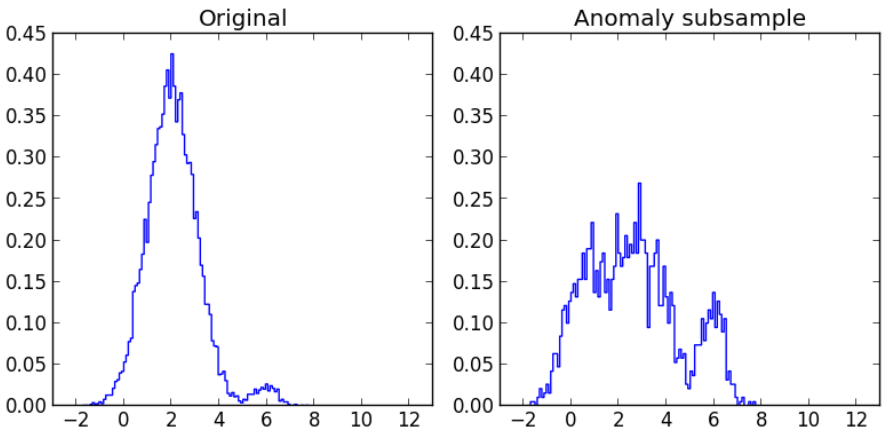

FIGURE 3.4: Effects of density based subsampling on the distribution of data. The left plot shows a normalized histogram of 10300 events split between two clusters, with 10000 events coming from the left hand cluster and 300 events from the right. The right plot shows the normalized histogram of 2000 random drawn events inversely proportional to the posterior probability of the event being drawn from the negative control. The relative frequency of the smaller cluster is increased, but the distribution of the larger population is distorted.

$$p(X) \propto \log\left(\frac{q(X|\theta_{\text{Positive}})}{q(X|\theta_{\text{Negative}})}\right) \tag{3.4}$$

where $q(X|\theta_{\text{Positive}})$ is the posterior probability of X being drawn from the positive control and $q(X|\theta_{\text{Negative}})$ is the posterior probability of X being drawn from the negative control. This causes events likely to come from the positive control but unlikely to come from the negative control to be highly likely to be randomly drawn for inclusion in the biased subsample.

In contrast to all the prior subsampling methods, interesting event subsampling preserves the negative distribution while increasing the frequency of rare events. Since uninteresting events are equally likely to be found in the negative and positive control they will have roughly a constant posterior probability ratio. This causes background events to be uniformly sampled and included in the subsample. This preserves the background distribution while increasing the proportion of rare events

as shown in Figure 3.5

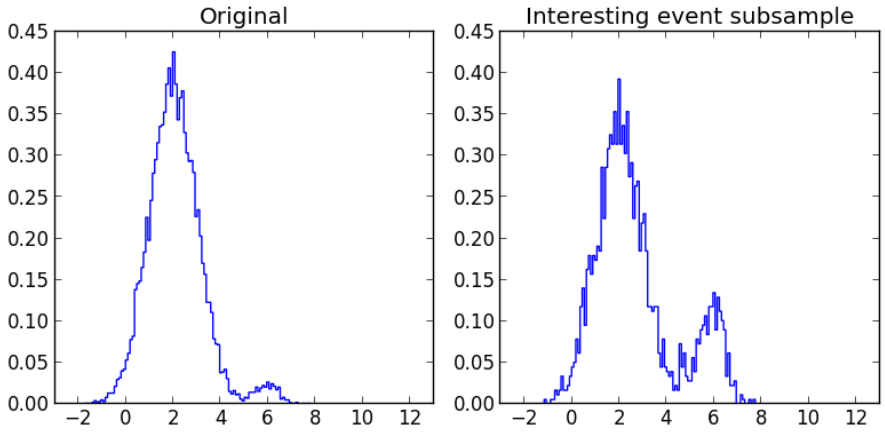

FIGURE 3.5: Effects of density based subsampling on the distribution of data. The left plot shows a normalized histogram of 10,300 events split between two clusters, with 10,000 events in the left cluster and 300 events in the right. The right plot shows the normalized histogram of 2,000 random drawn events proportional to the ratio of the posterior probability of the event being drawn from the positive control and the negative control. The relative frequency of the smaller cluster is increased, while the distribution of the larger population is preserved.

3.3.3 Determining sample sizes

One issue with subsampling is determining the sample size. If the sample size is too large only minimal computational improvement can be realized and masked events will remain hidden, but sample sizes that are too small may either contain too few events of interest or result in excessive bias in the fitted model. A simple heuristic for setting the sample size can be determined by using estimates of the relative frequency of the target cells of interest. The calculation of sample size assumes all cells of interest will be included in the subsample, and then sets those events to a predetermined frequency in the final subsample. If the number of events in a sample is n and the expected frequency of the cells of interest is f and final desired frequency is l the sample size s should be roughly

$$s \approx \frac{fn}{l} \tag{3.5}$$

For example with a desired final frequency of 1 event in 100, a predicted initial frequency of 5 in 10,000 and a total sample size of 100,000 events, our subsample size should be

$$\frac{0.0005 \cdot 100000}{.01} = 5000 \tag{3.6}$$

Assuming all target rare cells are selected the frequency will be $\frac{50}{5000}$ or 1%.

3.4 Results

3.4.1 Generation of synthetic data

Synthetic data was generated to illustrate uniform, density based, anomalous and interesting event subsampling. The synthetic data was constructed to represent detection of a rare cell population of interest. The negative control lacks this small population and a positive control contains a large population representing the rare cells. In the negative control 20,000 events were drawn from two normal distributions with means at (1,1) and at (5,1). Both distributions had variance 1, with 10,000 events coming from each distribution.

$$\begin{aligned} X_{1\cdots10000} &\sim N\left(\mu = \begin{bmatrix}1\\1\end{bmatrix}, \Sigma = \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}\right) \\ X_{10001\cdots20000} &\sim N\left(\mu = \begin{bmatrix}5\\1\end{bmatrix}, \Sigma = \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}\right) \end{aligned} \tag{3.7}$$

Similarly, for the positive control, 30,000 events were drawn from three normal distributions, with means at (1,1), at (1,5), and at (5,5), all with variance 1. 10,000 events coming from each of the three distributions.

$$\begin{aligned} X_{1\cdots10000} &\sim N\left(\mu = \begin{bmatrix}1\\1\end{bmatrix}, \Sigma = \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}\right) \\ X_{10001\cdots20000} &\sim N\left(\mu = \begin{bmatrix}5\\1\end{bmatrix}, \Sigma = \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}\right) \\ X_{20001\cdots30000} &\sim N\left(\mu = \begin{bmatrix}5\\5\end{bmatrix}, \Sigma = \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}\right) \end{aligned} \tag{3.8}$$

For the sample data set 20,000 events were drawn from two normal distributions with means (1,1) and (5,5) and variance 1, matching the negative control. In addition, 20 events, representing the cells of interest, were drawn from a normal distribution with mean (5,4.5) and variance 0.5. The cells of interest have a lower mean in the Y axis as often positive control have a much stronger response than treatment groups.

$$\begin{aligned} X_{1\cdots10000} &\sim N\left(\mu = \begin{bmatrix}1\\1\end{bmatrix}, \Sigma = \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}\right) \\ X_{10001\cdots20000} &\sim N\left(\mu = \begin{bmatrix}5\\1\end{bmatrix}, \Sigma = \begin{bmatrix}1 & 0\\0 & 1\end{bmatrix}\right) \\ X_{20001\cdots20020} &\sim N\left(\mu = \begin{bmatrix}5\\4.5\end{bmatrix}, \Sigma = \begin{bmatrix}0.5 & 0\\0 & 0.5\end{bmatrix}\right) \end{aligned} \tag{3.9}$$

If we assume our models have a detection limit of 2%, using Equation 3.5 we then calculate a sample size of 1,000 events.

Since the true distribution contains data from 3 normals, we model the data set using a mixture of 3 normal distributions. Models were fitted to the data via Markov chain Monte Carlo, with 1,000 burn-in iterations, and 100 posterior draws that were averaged.

3.4.2 Uniform subsampling

As shown in Figure 3.6, uniformly subsampling 1,000 events from the original 20,020 events manages to preserve the gross structure of the two larger modes, but does not increase the ability to detect the 20 cells of interest with a three component Gaussian mixture model. In fact the probability that none of the 20 cells of interest is drawn is approximately 36%.

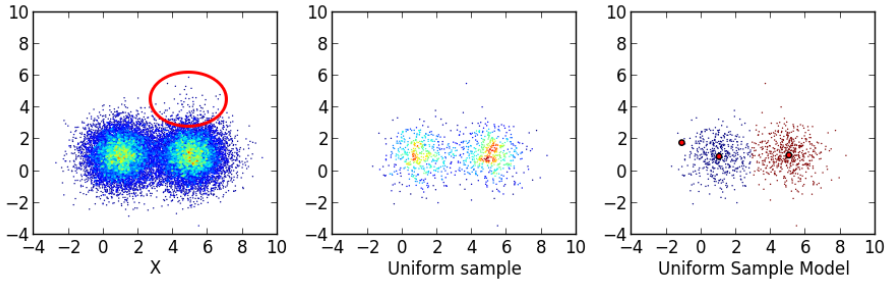

FIGURE 3.6: Illustration of uniform subsampling. The sample comprises 20,020 events, and the uniform sample consists of 1,000 events drawn from the original 20,020. The red circle highlights the small population of rare events. Fitting a mixture model to the uniform sample fails to properly identify the small cluster of 20 events as they are likely not present in the sample. Left panel show a plot of the original data set. The center panel shows a plot of 1,000 events uniformly subsampled from the data set. The right panel shows the classification of the 1,000 sampled events by a three component mixture model.

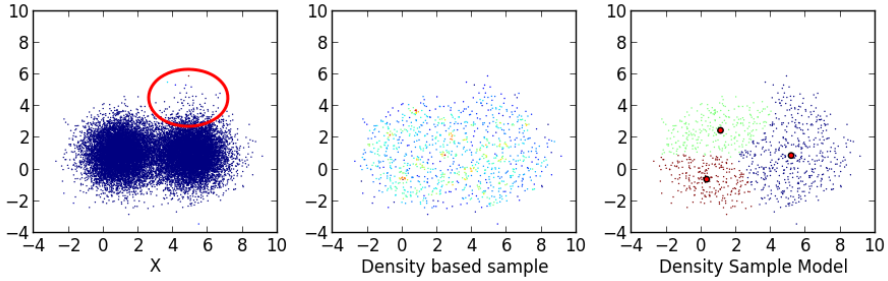

FIGURE 3.7: Density based biased subsampling. Events from low density regions are included at a higher frequency than events from high density regions, resulting in a more uniform looking distribution to the subsample. Left panel show a plot of the original data set. The red circle highlights the small rare event population. The center panel shows a plot of 1,000 events subsampled via density based biased subsampling as described in the text. The right panel shows the classification of the 1,000 events by a mixture model with three components fitted to the subsample.

3.4.3 Density based subsampling

With density based subsampling, events from low density regions are over represented when compared with events from high density regions, resulting in a sample with a flatter looking distribution, as seen in Figure 3.7. While the small population of rare cells of interest appears to be present in the sample, the more uniform distribution of the subsample masks the cells of interest, and prevents them from being easily identified as a distinct cluster by model based methods. To estimate this flatter distribution, more mixture components would be needed to accurately describe the data set, increasing the computational run-time. Furthermore, it is unlikely that the model would identify the rare population, as the demarcation between the rare population and the background events is reduced.

3.4.4 Anomalous event subsampling

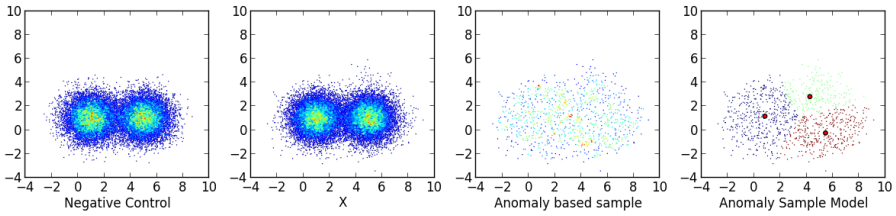

FIGURE 3.8: Subsampling looking for events unlikely to be from the distribution of the negative control. While the events of interest appear to be included in the sample, so do many of the low probability events in regions not of interest. The left panel shows a plot of the negative control. The left-center plot shows a plot of X, the original data set. The right-center panel shows a plot of 1,000 events sampled from the original data set X using anomalous subsampling. The sample was biased to prefer events that had a low probability of coming from the negative control. The right panel shows the classification of the 1,000 sampled events as classified by a mixture model with three components fitted to the sample.

Anomalous event subsampling uses the negative control to identify anomalous events. An example of anomalous subsampling is shown in Figure 3.8. Anomalous subsampling preserves the rare population, but similar to density based subsampling,

flattens the high density regions. Hence it suffers the same problems as density based subsampling and will likely require more components to accurately describe it, as well as causing rare events to be masked even more.

3.4.5 Interesting event subsampling

A synthetic data set comprising a test sample with a positive control and negative control are used to illustrate interesting event subsampling in Figure 3.9. After subsampling 1,000 events, a three component mixture model places a component in the region of the events of interest. By using the positive and negative control it becomes possible to correctly identify the small masked component, with minimal distortion of the background distribution. Despite identifying the three components, there is significant bias in the estimated model as can be seen by the classification of the original data set in the bottom left plot of Figure 3.9.

Compensating for bias in the estimated model

One consideration when using biased samples is that they increase the likelihood of incorrectly identifying the rare population. Despite identifying the three components, there is significant bias in the estimated model as can be seen by the classification of the original data set in the bottom left plot of Figure 3.9. Because the model is fit to biased data, it may not describe the original data set accurately, as seen in the bottom-left plot in Figure 3.9. Many events from the negative population are misclassified as belonging to the small positive population.

To avoid this problem, the model can be used to set prior means, co-variances and proportions to a new model instead of using it to classify events. Using this mixture model to provide prior weights, means and covariances for a new mixture model fit to the original data set increases the likelihood of correctly identifying the rare population.

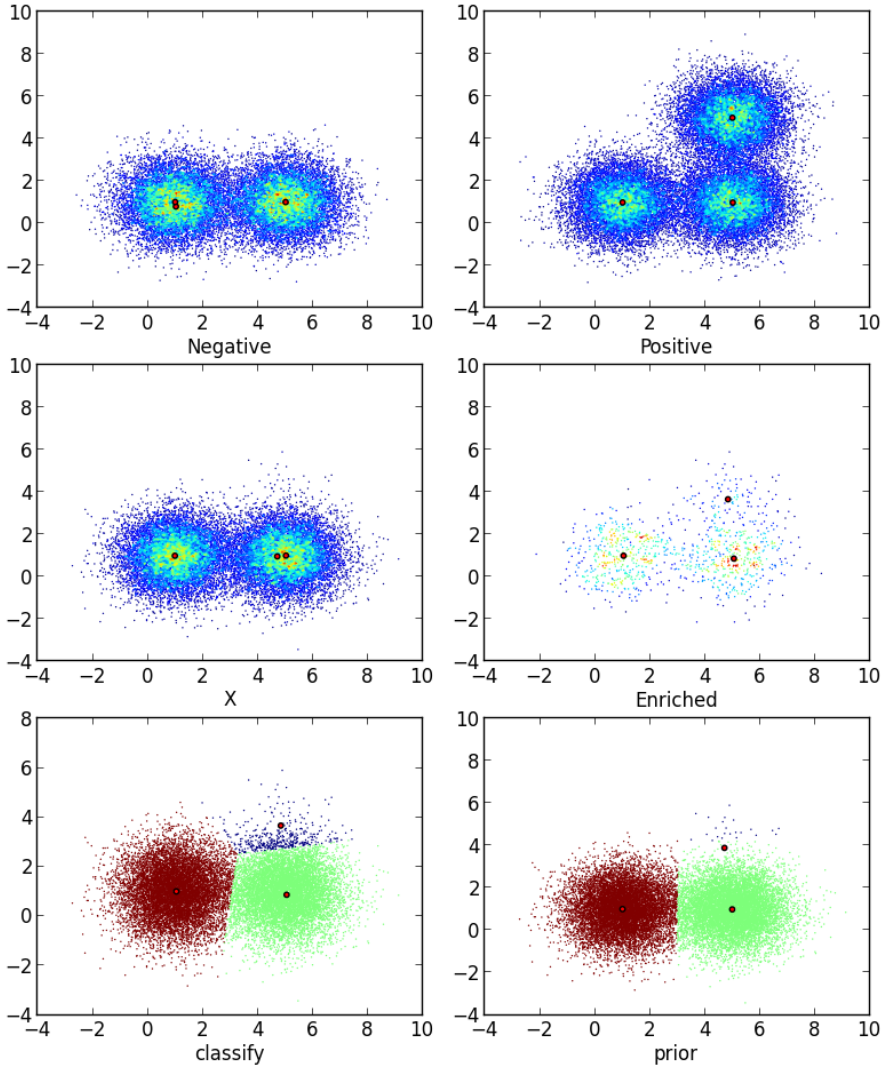

FIGURE 3.9: Example of biased subsampling on a synthetic data set. The top-left plot is the negative control, consisting of 20,000 events drawn equally from two Gaussians, with means (1,1) and (5,1), and variance 1. The top-right plot is the positive control, consisting of 30,000 events drawn from three Gaussians, with means (1,1), (5,1), and (5,5) all with variance one. The center-left plot is our data sample X, 20,000 events drawn from two Gaussians with means (1,1) and (5,1) and variance 1, and 20 events from a normal with mean (5,4.5) and variance 0.5. The center-right plot is a 1,000 event biased sample of X. The bottom-left plot is the classification of events in X using a mixture model fit to the biased subsample. The bottom-right plot is the classification of X using a mixture model with prior means, proportions and co-variances set by the mixture model fit to the biased subsample. Larger red dots are means of a 3 component mixture model fit to the data plotted, except in the bottom-left plot, which uses the same mixture model fitted in center-right plot, the biased subsample.

3.4.6 Interesting event subsampling on spiked data

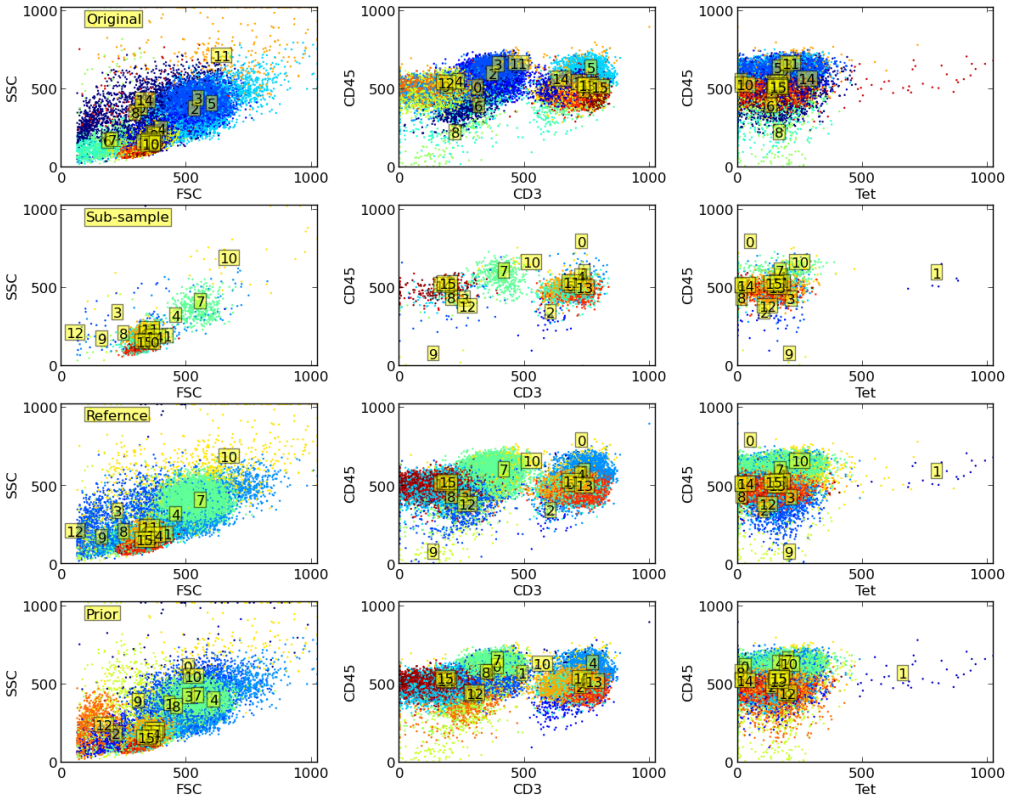

FIGURE 3.10: Detection of rare events using biased subsampling. Using 16 components in the original mixture model model it is difficult to detect the rare tetramer population. By biased subsampling down to 2,500 event the population becomes easier to detect. A mixture model fit to the biased sample is used as a reference to classify the original data, and generate prior means, proportions and co-variances for a new mixture fit to the original data. Top row shows a mixture model of 16 Gaussians fit to a sample spiked with 0.013125% tetramer positive cells. The second row shows a mixture model of 16 Gaussians fit to a 2,500 event biased sample. The third row shows the use of the model fit to the sample as a classifier for the original data set. The fourth row shows a 16 component Gaussian mixture model fit to the original data using the subsample to generate prior means, proportions and co-variances.

To evaluate the utility of interesting event subsampling on experimental data, we used a set of data samples "spiked" with a known frequency of TCR-transfected antigen specific T cells, specific the NY-ESO-1 canter-testis antigen (Singh et al., 2013). The spiked concentrations of antigen specific T cells used are 0%, 0.013125%,

and 0.22%. There is also an estimated 0.01% contribution by background antigen specific T cells in the samples prior to being spiked as estimated by expert gating.

For interesting event subsampling the 0% spiked population sample was used as a negative control, and the 0.22% spiked population sample was used as a positive control. Since the frequency of spiked antigen specific cells in the sample was 0.013125% of the 50,000 events, we subsampled 2,500 events, which should bring the rare antigen specific events to roughly 0.4625% of the subsample. The rare tetramer positive population in the 0.013125% sample was not found with a Dirichlet process Gaussian mixture model with 16 components, but was found in the biased sample. Using the mixture model fit to the biased sample, the tetramer positive population is identified as cluster 1 (see Figure 3.10), which comprises 0.24% of the events in the sample. Using the mixture model fit to the sample identifies a cluster which comprises 0.04% of events in the sample. Using the model fit to the sample to provide priors to for a new mixture model finds a component that comprises 0.072% of events in the sample.

3.5 Discussion

Many subsampling approaches enrich for rare events potentially improving computational efficiency. However, each of the different subsampling methods can distort the distribution of samples, making further analysis difficult by masking, loss of rare populations, or requiring more components to describe the distribution. While one goal is to increase the frequency of rare events to ease detection, care must be taken not to distort the distribution of these events, or of the distribution of non-rare events. Figure 3.11 illustrates how the different methods can change the distribution of both rare and non-rare events.

Random subsampling maintains the "shape" of the distribution of both rare and non-rare events, but fails to increase the frequency of rare events, and hence does

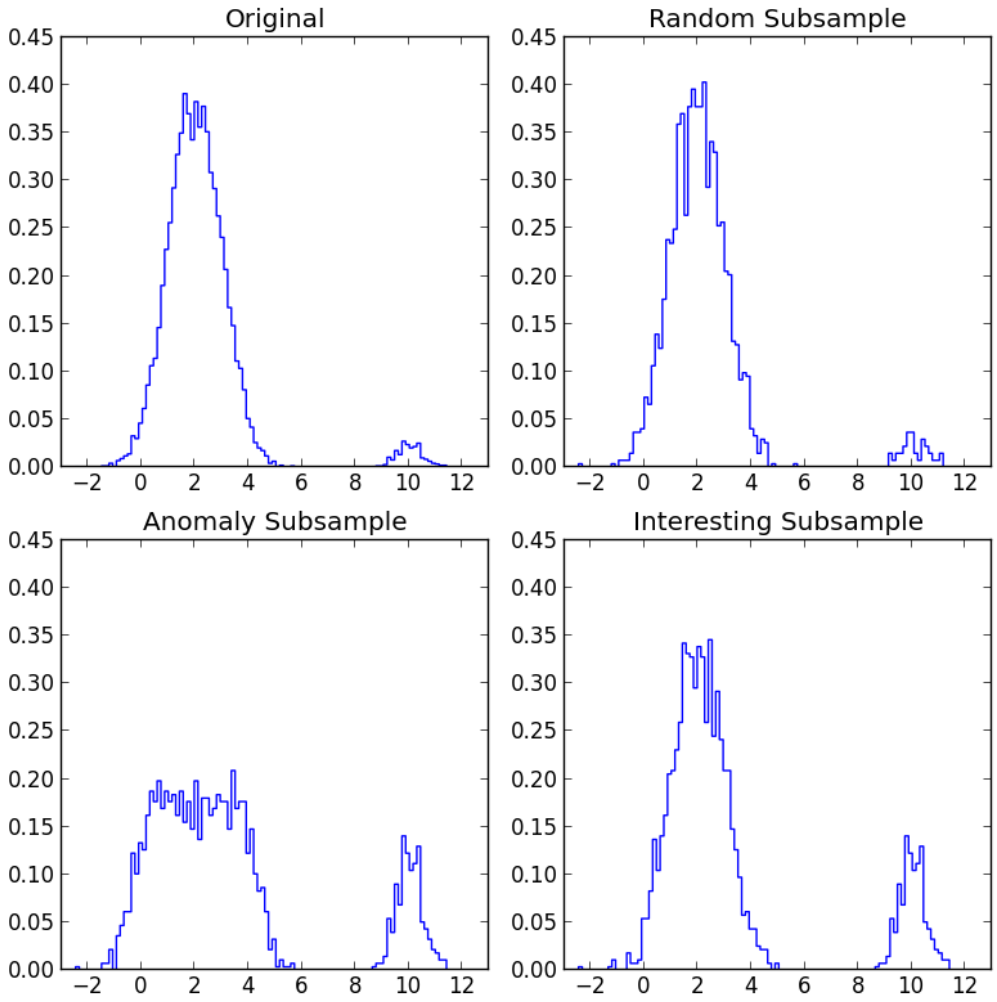

FIGURE 3.11: Illustration of how various subsampling methods distorts distributions. Top left is the original sample consisting of 10300 events, with the larger 10,000 event population drawn from a normal with mean 2 and variance one, and the smaller 300 event population drawn from a normal with mean 10 and variance 0.5. Top right shows random subsampling of 2,000 events. Random subsampling does not significantly change the distribution. Bottom left panel shows anomaly subsampling of 2,000 events. Anomaly subsampling increases the over all frequency of the rare population, but can significantly distort the distribution of the larger population. The bottom right panel shows biased subsampling of 2,000 events. Biased subsampling increases the frequency of the smaller population without significant changes to the larger population.

not aid in their detection and, if the sample size is too small, it could remove the rare populations entirely. Density based and anomaly subsampling increases the frequency of rare events and maintains their distribution, but can significantly distort the distribution of non-rare events. This shift in distribution can cause problems for model based analysis, as a single population may now need multiple components to accurately describe the new distorted distribution. With good positive and negative controls, interesting event subsampling can increase the frequency of rare events, while maintaining the "shape" of the distribution of both rare and non-rare events.

Even so, interesting event subsampling does create biased samples. The models estimated from the subsample may not be representative of the true distribution of the original sample. As seen in Figure 3.9, this can cause unwanted events to be included in rare populations when using the estimated model to classify the original data set. Many of the events on the boundary between the rare population and the nearby negative population were misclassified. Because of this bias, it will often be more accurate to use the biased model to set priors for a new model fitting as was done in the examples.

While interesting event subsampling offers many advantages over other subsampling methods, it does require a positive control, which is not run for every experiment. In addition, it requires the positive control to be representative for the target rare cells. Novel cell populations are unlikely to be found unless they are well represented in the positive control. In addition, interesting event subsampling will increase the frequency of real but unwanted differences, such as dead cells. Activated T cells have increased cell turn over. Since the negative control lacks this activation, these dead cells will be represented in both the treatment and positive control, causing them to be over represented in the biased sample. Typically this is not a problem, as dead cells tend to form distinct clusters, and are easily excluded. Use of viability dyes, and other exclusion markers to "pre-gate" samples could also be used

to mitigate this effect.

Two alternatives to subsampling for reducing the problem space are to partition either the marker space, or the events. We can perform model fitting in sequential stages and only include events relevant at that stage. Similar to the iterative process of gating, first T cells are identified using scatter, viability dyes and CD3 channels, and then from those T cells identify helper T cells using CD4 and CD8 channels. Since dimensionality is reduced and the number of events evaluated with each stage is smaller the run-time can often be shorter. However, marker partitioning runs into many of the same problems encountered in gating, as it requires expert knowledge of what sub-spaces to choose at each iteration and which events should be considered for the next round of modeling.

Similar to marker partitioning, it is possible to partition regions of fluorescent space and only examine the events in that region. This means the model only needs to consider a fewer number of events, and can ignore complexities in regions of space it is not examining. However, this also requires expert knowledge of the cell subsets of interest. A further problem with partitioning in this way is that it can induce artificial boundaries. If a partition boundary between regions cuts a population in half, frequencies estimated will be erroneous. Sophisticated statistical methods to accurately partition markers and events have been developed (Lin et al., 2013), but are complex and relatively computationally inefficient compared to standard Gaussian mixture models.

4

Identifying Common Populations Across Samples

4.1 Introduction

Cross sample comparison is needed for many flow studies, as many flow cytometry experiments involve comparing different samples to each other. For example case-control studies are interested in differences between populations in the case and control samples; longitudinal studies are interested in tracking changes in populations over time; multi-center clinical trials will have flow cytometry samples run in different labs that would need to be compared to the other labs. In most typical applications of statistical approaches to flow cytometry analysis, models are fitted to samples independently. Consequentially population 1 in the first data set may not represent the same population labeled 1 in a second data set. Ultimately, the cluster labels in a fitted model are arbitrary with respect to another model fitted independently. If clusters were relabeled to be consistent across samples, so that cluster labels for the sample populations are the same across samples, cross sample comparison would be significantly facilitated (see Figure 4.1).

One of the simplest methods to achieve consistent cluster labeling is to use a

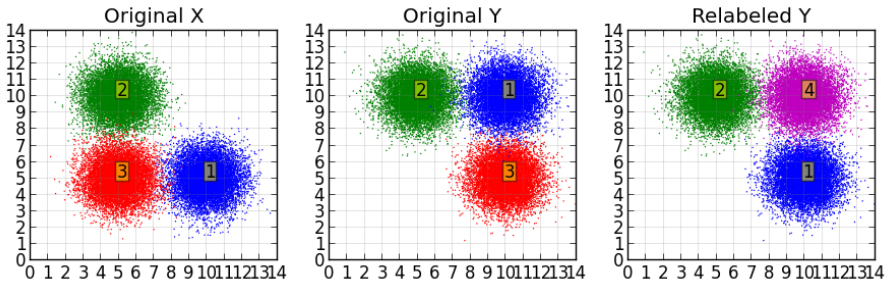

FIGURE 4.1: The goal of cluster alignment is to find common clusters between two arbitrarily labeled data sets, and relabel them such that clusters common to both data sets have the same label, while clusters unique to each data set are numbered uniquely. The original data sets share two common clusters, labeled 1 and 2 in X, and 2 and 3 in Y as seen in the middle plots. After relabeling the common components are share the same label across both data sets while component 3 is unique to data set X and component 4 is unique to the relabeled data set Y as shown in the right plot.

model fit to a reference data sample to assign cluster membership. One of the samples is nominated to be the reference, and a model is generated from that reference file. This model is then used to classify the remaining data sets. While easy to implement, it can run into significant problems if the other data sets contain populations not present in the original reference sample, as seen in Figure 4.2.

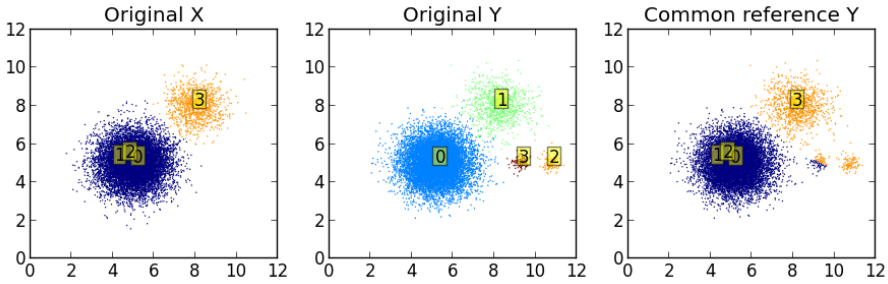

FIGURE 4.2: Consistent labeling by using a reference data set. Left plot shows the data set X, with two populations. The middle plot shows the second data set, Y, before classification by a model fit to data set X. The right most plot is of data set Y classified by a model fit to data set X. The two small populations are misclassified as no corresponding population exists in X.

A simple method to overcome the missing population problem in using a reference

model is to pool all the data sets and fit a single model to the pooled data. While simple to perform, pooling data is problematic with a large number of samples. The pooled data set can become too large to fit feasibly in computer memory with hundreds of samples. To counter the increased size, down-sampling needs to be performed, leading to the potential to miss rare populations. In addition, pooling may result in the incorrect merging of overlapping but distinct populations. As seen in Figure 4.3, the two small populations labeled 2 and 3 in the Y sample end up being merged into a single population when fit with a common model from a pooled sample of 2,000 events drawn from each sample. The small populations are likely merged due to subsampling not including enough events to form distinct clusters. Pooling also may not be feasible if intermediate analysis is needed on samples, as may be the case in longitudinal studies. Ultimately both reference and pooling suffer from the same problem: the same model is used to classify potentially heterogeneous samples.

A different approach to relabel cluster labels is to cluster the clusters. Pyne et al. (2009) used the partitioning around medoids algorithm to cluster the cluster means to identify cell subsets in the FLAME method. Partitioning around medoids requires knowledge of the optimal number of medoids for clustering, and all the data samples have to be available at the same time. It may also merge multiple clusters in a data set into a single cluster as seen in Figure 4.4, cell subsets labeled 4 and 3 in the Y sample are merged into a single population.

Another method of clustering clusters is agglomerative hierarchical clustering. In agglomerative hierarchical clustering, individual clusters are merged based on some similarity measure until only a single grand cluster remains. Typically distance measures, such as Euclidean distance, are used for the measure of similarity. An example of agglomerative clustering is shown in Figure 4.5. Using agglomerative clustering it is possible to merge clusters in the same data set rather than relabel

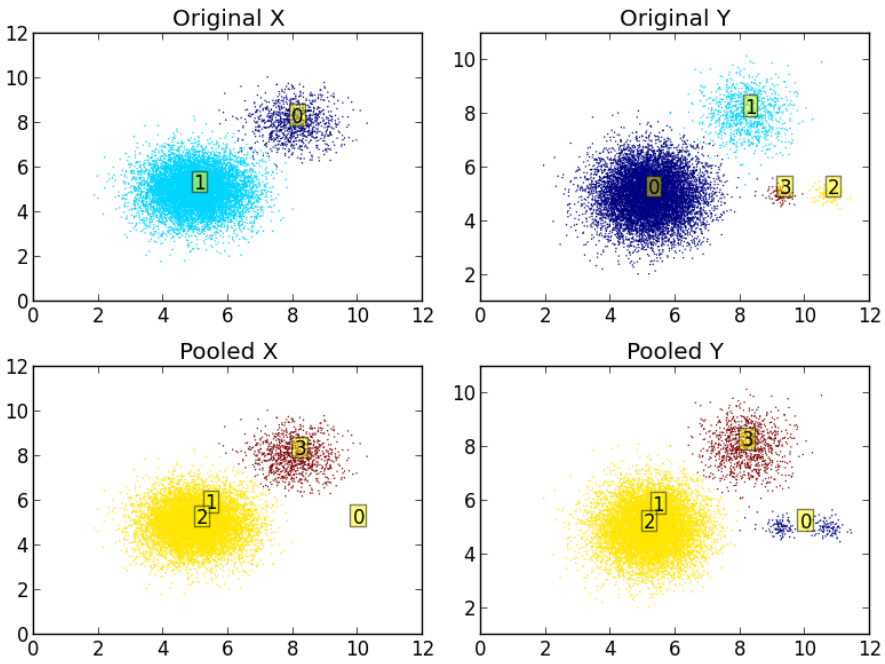

FIGURE 4.3: Pooling to provide common labeling across data sets. Top left shows the original X population before relabeling, consisting of two populations. Top right shows the original Y population before relabeling consisting of two large populations and two small populations. The bottom left shows the original X classified by a model fit to a pooled sample comprising 2,000 events drawn randomly from both X and Y. Similarly the bottom right plot shows the Y sample classified by the same model as in the bottom left.

them. The final number of clusters is determined by setting a stopping value for agglomerative clustering. Using the example in Figure 4.5, setting a cut off value of 1 will result in four clusters, while choosing a cut off of 2 results in three clusters. However, the value of the cut of is typically both ad-hoc and subjective, chosen after visual inspection of the resulting dendrogram.

One more solution for maintaining consistent labels across data sets is to employ hierarchical models as proposed by Cron et al. (2013). This method uses a hierarchical Dirichlet processes (HDP) to generate an individual Gaussian mixture model for each sample, but ensures that mixture models fit across samples share common means an covariances, while allowing them to have independent weights. HDP is not

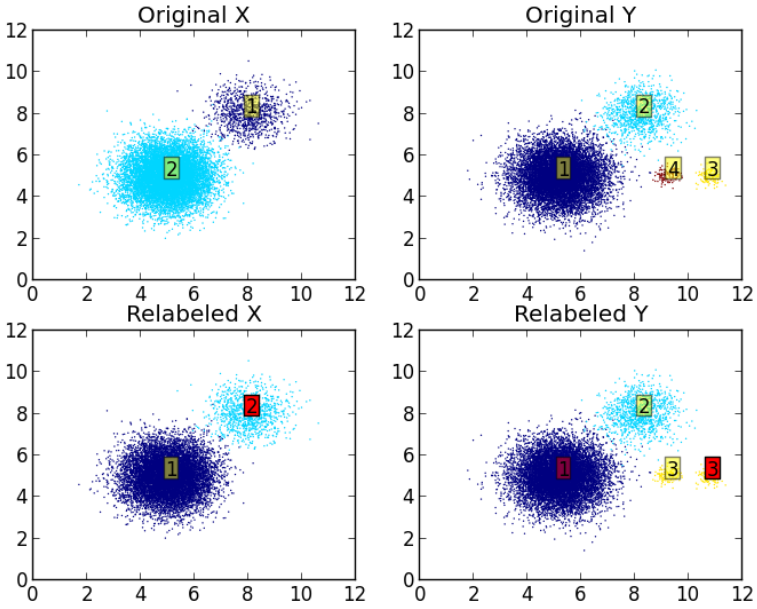

FIGURE 4.4: Partitioning around medoids to relabel clusters. Top left shows the original X clusters before relabeling. Top right shows the original Y clusters before relabeling. Bottom left shows the relabeled X and bottom right shows the relabeled Y. Red labels are the chosen medoids.

a relabeling technique but a different type of model. As shown in Figure 4.6, HDP keeps labels consistent across samples, and can manage missing populations. Similar to pooling data, HDP requires all data sets to be available at the time of fitting, which may not always be available.

We propose a new method of relabeling, based on the Munkres assignment algorithm, where only two data samples need to be analyzed at any give time, offering the advantage of incremental updates. This is an important consideration when data is collected in a longitudinal fashion, as this allow interim analysis. In the next section, we explore how the Munkres algorithm can be used to relabel models.

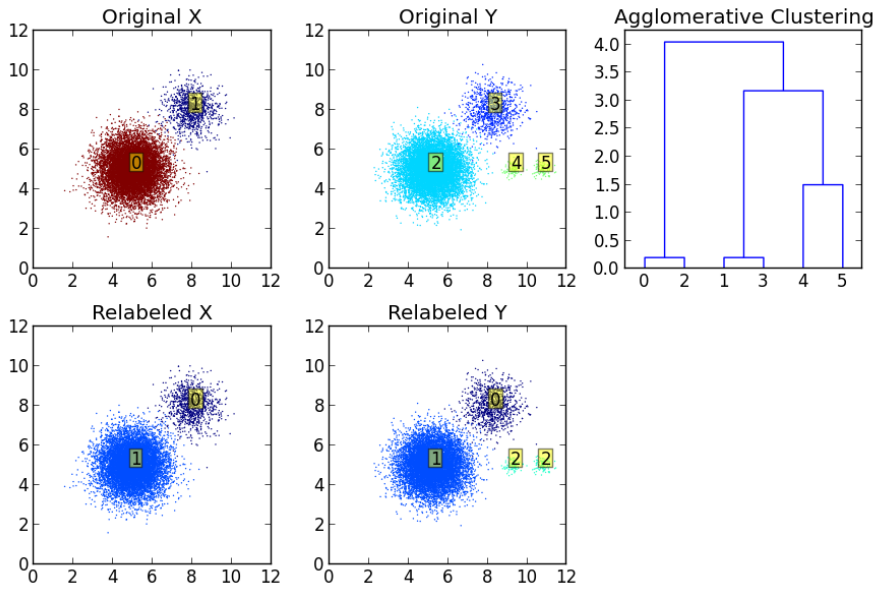

FIGURE 4.5: Agglomerative hierarchical clustering to relabel clusters. Top left plot shows the original X sample before relabeling. Top center shows the original Y before relabeling. The top right shows the dendrogram of the order of cluster merging. At a cut off of 2 only three clusters remain. The bottom left shows the relabeled X and the bottom right plot shows the relabeled Y.

4.2 Algorithm

A successful relabeling algorithm needs to be able to match clusters across samples, while handling missing or new populations. We propose a method utilizing the Munkres algorithm for solving the assignment problem to handle the relabeling of clusters, and extend it to account for new populations. The Munkres algorithm was previously employed to relabel clusters across MCMC iterates by minimizing misclassification by Cron and West (2011), solving to the label switching problem. Here we apply the algorithm to data sets in a new context, that of relabeling data sets fit independently by Gaussian mixture models.

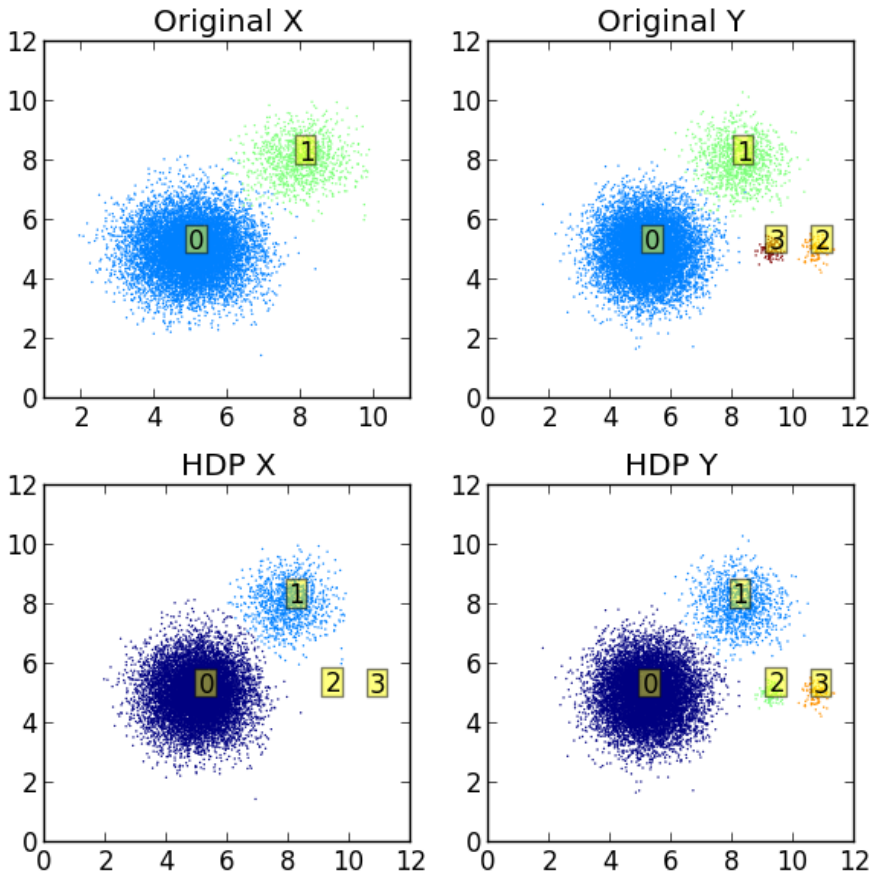

FIGURE 4.6: Clustering with HDP. The top left plot shows a data set labeled X before being reclustered with HDP. The top right plot shows data set Y. The bottom row shows X on the left and Y on the right after being clustered with HDP. The resulting cluster labels are consistent across samples, even when there are missing populations as shown on the bottom left plot of X.

4.2.1 Munkres

The Munkres algorithm, also known as the Hungarian algorithm, is a classical computer science algorithm for solving the assignment problem. In the assignment problem, agents are assigned individually to a task from a set of tasks, with each agent having a cost associated with performing each given task (an example is provided in Table 4.1). No agent is assigned to more than one task. A matrix of costs, C is generated, where $C_{i,j}$ is the cost for agent i to perform task j. The Munkres algo-

Table 4.1: Example of the results of the Munkres algorithm. The optimal cost assignment would be the assigning C to x, B to y and A to z.

	x	y	z
A	1	2	3
B	5	6	9
C	9	11	13

rithm finds a minimal cost solution to the assignment problem in $O(n^3)$ time and is far more efficient than brute force methods that examine every possible assignment in $O(n!)$ time.

For the problem of assigning cell subsets across samples, we treat the features of the posterior distribution in a reference data set as the agents, and the features of the posterior distribution in the second data set as the tasks. Using the modes of the distributions as features, the cost matrix is then composed of some measure of dissimilarity between the modes in the reference data set and the modes in the second data set. Typical measures of dissimilarity include Euclidean distance, Kullback-Leibler divergence, and misclassification rate between features.

4.2.2 Extension to non-square cost matrices

The common implementation of the Munkres algorithm assumes a one-to-one correspondence between agents and tasks. This constrains the cost matrix to be a square matrix. However, we often need to align samples where the number of modes is unequal. The common solution for the Munkres algorithm to handle mismatch between the number of agents and tasks is to pad the cost matrix with dummy agents or tasks until the number of each set is equal with the costs in the added rows or column set to zero. The dummy agents and tasks added to the cost matrix are then ignored when looking at the resulting assignments. Bourgeois and Lassalle (1971) provide a modification of the Munkres algorithm that performs better than simply padding the cost matrix. Using this modification allows for the relabeling of samples

Table 4.2: Example of the the Munkres algorithm with maximum cost. The cost matrix has been padded with three additional columns. The optimal cost assignment would be to not assign C to any task, B to x and A to y.

	x	y	z			
A	1	2	3	6	6	6
B	5	6	9	6	6	6
C	9	11	13	6	6	6

with unequal numbers of modes.

4.2.3 Max cost extension

Even when the number of modes in a data set is the same, it is not necessarily the case that every mode in the reference has a corresponding mode in the other sample. We, therefore, developed a method of breaking "poor" assignments that still respects the generation the minimum cost of assignment. With the addition of the ability to handle non-square cost matrices, breaking poor assignments becomes possible by generating a number of additional dummy tasks as shown in Table 4.2. By padding the cost matrix with additional dummy tasks equal to the number of agents, all with a fixed cost, we stop any agent from being assigned a task with a greater cost than a dummy task. Hence with N modes in the reference data set and M modes in the sample, our cost matrix has dimensions $N \times M + N$, with the last N columns being filled with the maximum cost value we wish to see in the final assignment. Constructing the cost matrix as such, the maximum value of assigning one mode to another with the minimal cost will be the assignment of a mode to one of these dummy modes. Hence the cost of any assigned modes in the the first M columns are less than or equally as expensive as assigning to a dummy mode. Using this, we can constrain the method to only assign modes below a maximum cost. Any modes in the reference data set assigned to a dummy mode are assumed then to have no corresponding mode in the other sample.

4.2.4 Dissimilarity measures

The cluster relabeling method proposed uses one of several measures of dissimilarity. The first measure of dissimilarity is Euclidean distance, defined as

$$d = \sqrt{\sum_{i=1}^{n} (p_i - q_i)^2} \tag{4.1}$$

where p and q are the centers of the two modes being compared. While easy to calculate, the Euclidean distance does not account for differences in the shapes of the two modes.

Another measure of dissimilarity between two components is the misclassification rate. Misclassification can be easily implemented by counting the number of events classified to the mode in the second mixture model when the events are drawn from the a given mode in the first mixture model. Misclassification can account for differences in shape, but runs into issues with missing populations as every event has to be classified to some mode.

The third measure of dissimilarity is KullbackLeibler divergence, a non-symmetric measure of the differences between two distributions defined previously in Equation 2.3 and restated here

$$D_{KL}(P\|Q) = \int_{-\infty}^{\infty} \log \frac{P(x)}{Q(x)} P(x)\, \mathrm{d}x \tag{4.2}$$

which in the case of the difference between two k dimensional Gaussian distributions simplifies to

$$D_{KL}(P\|Q) = \frac{1}{2}\left(\mathrm{tr}\left(\Sigma_Q^{-1}\Sigma_P\right) + (\mu_Q - \mu_P)^\top \Sigma_Q^{-1}(\mu_Q - \mu_P) - k - \ln\left(\frac{\det \Sigma_P}{\det \Sigma_Q}\right)\right) \tag{4.3}$$

4.3 Results

Relabeling was performed on a variety of synthetic data sets to illustrate some of the limitations of each dissimilarity measure, and to illustrate the need to assign a maximum cost of relabeling. All three dissimilarity measures were also performed on a subset of the EQAPOL data set, both with and without a maximum cost in each case.

4.3.1 Synthetic data

A variety of synthetic data sets with common, new, and missing components were used. A perfect alignment algorithm will find common components between data sets while correctly identifying new and missing populations.

The reference sample, X, consists of two equally sized components with mean $(5, 5)$ for component 1 and mean $(10, 5)$ for component 0. The test sample, Y, consists of three components. Component 0 in Y has mean $(5, 5)$, component 2 has mean $(10, 5)$, and component 1, the smaller of the three, has mean $(10, 9)$. Figure 4.7 shows how the relabeling algorithm generates new labels for populations in the test sample, Y. Since component 1 in the test sample has no analog in the reference sample, it is assigned a new label when relabeled.

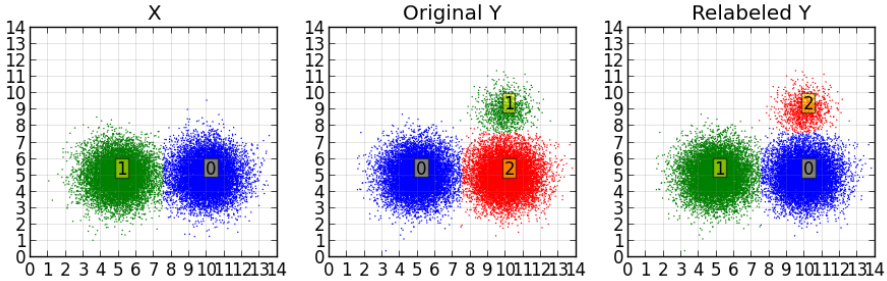

FIGURE 4.7: Alignment of samples with unequal numbers of clusters. Population Y has a double positive population, labeled 1, that does not exist in X. Relabeling Y assigns matching populations to the label in X, matching population 0 to 1 and 2 to 0. The new population then is labeled an unused population label, 2.

Relabeling has the potential to mislabel populations as being the same population due to the forced one-to-one assignment if the maximum cost is not used. Figure 4.8 illustrates this issues with the forced one-to-one assignment. The reference sample consists of three modes, at $(5, 5)$, $(10, 5)$, and $(10, 9)$. The test sample also consists of three modes at $(5, 5)$, $(10, 5)$, and $(5, 9)$. Because two of the modes align very well with each other, they correctly relabel, but the final mode is left with only one assignment option left. Despite the bad fit of this assignment, the algorithm relabels the mode as seen in the bottom left panel. By providing in a maximum allowable assignment cost, this final mode in the Y sample is correctly seen as having no match in the reference data set and being assigned a new unused label.

Using the different dissimilarity measures can result in different relabelings. Figure 4.9 illustrates one of the shortcomings of using Euclidean distance between means, and the need to consider the shape and the weight of the distribution features when relabeling. The reference sample X consists of two modes, with mean $(5, 5)$ and $(10, 5)$, while the test sample Y consists of three modes with means $(5, 5)$, $(10, 5)$, and $(12, 5)$. Euclidean distance between means assigns cluster 1 in the test sample to cluster 0 in the reference sample. Using a different measure of dissimilarity such as the Kullback-Leibler divergence, assigns cluster 2 in the test sample to cluster 0 in the reference sample instead.

4.3.2 EQAPOL

Cluster relabeling was run on data from the EQAPOL proficiency testing data sets. The EQAPOL data sets consist of 27 samples from 12 labs. The 27 samples from each lab consist of 9 samples from three subjects. The three subjects were common across all labs. Each of the samples from each subject included two stimulation samples and a brefeldin only negative control sample. Each of these samples has two technical replicates. A total of 324 samples were processed. Subjects were labeled

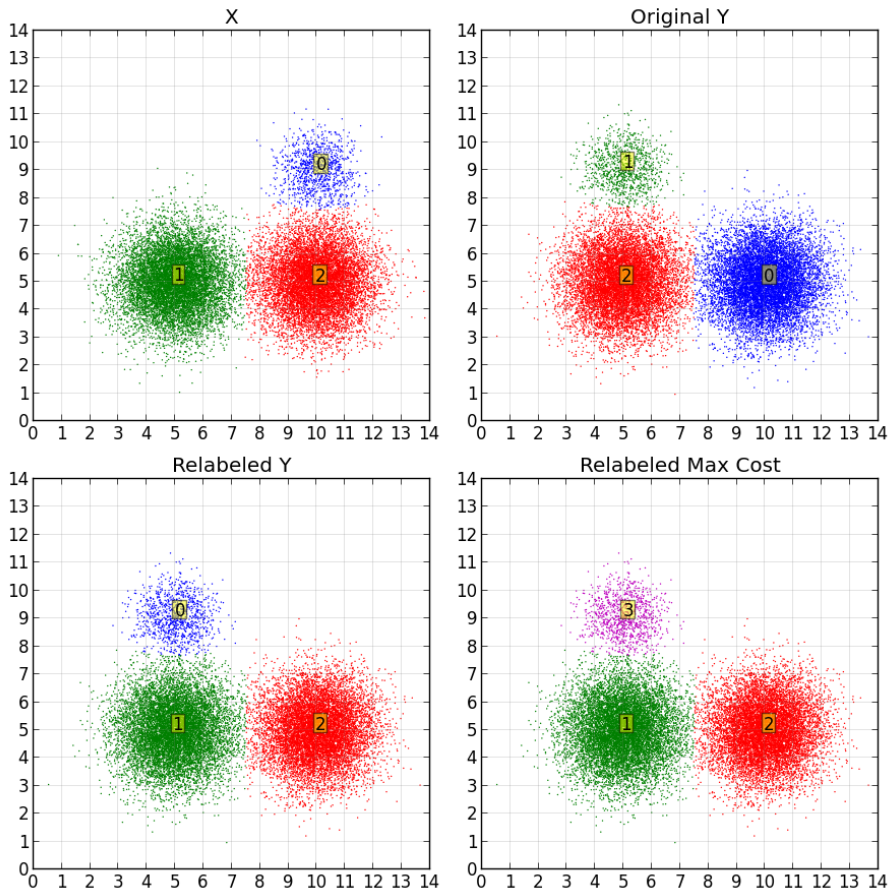

FIGURE 4.8: One to One assignment using the standard Munkres assignment algorithm can lead to mislabeling of some populations. Without a cost threshold, population 1 in Y is incorrectly assigned to population 0 in X. Setting a maximum allowable cost correctly assigns a new label.

A,C, and E. Samples number 1,2 and 3 are the brefeldin only negative control and technical replicates. Samples numbered 5,6,7 are a cytomegalovirus peptide pool stimulation (CMV pp65) sample and technical replicates. Samples numbered 9,10,11 are a cytomegalovirus, Epstein-Barr virus and influenza virus(CEF) epitopic peptide stimulation and technical replicates. Relabeling was performed across each of the samples from the subject labeled E. A 32 component Gaussian mixture model was fit to each data set. Between 18 and 21 modes were found in each sample. A CMV stimulation sample from the third subject labeled G6904VJT-07_E06 correctly

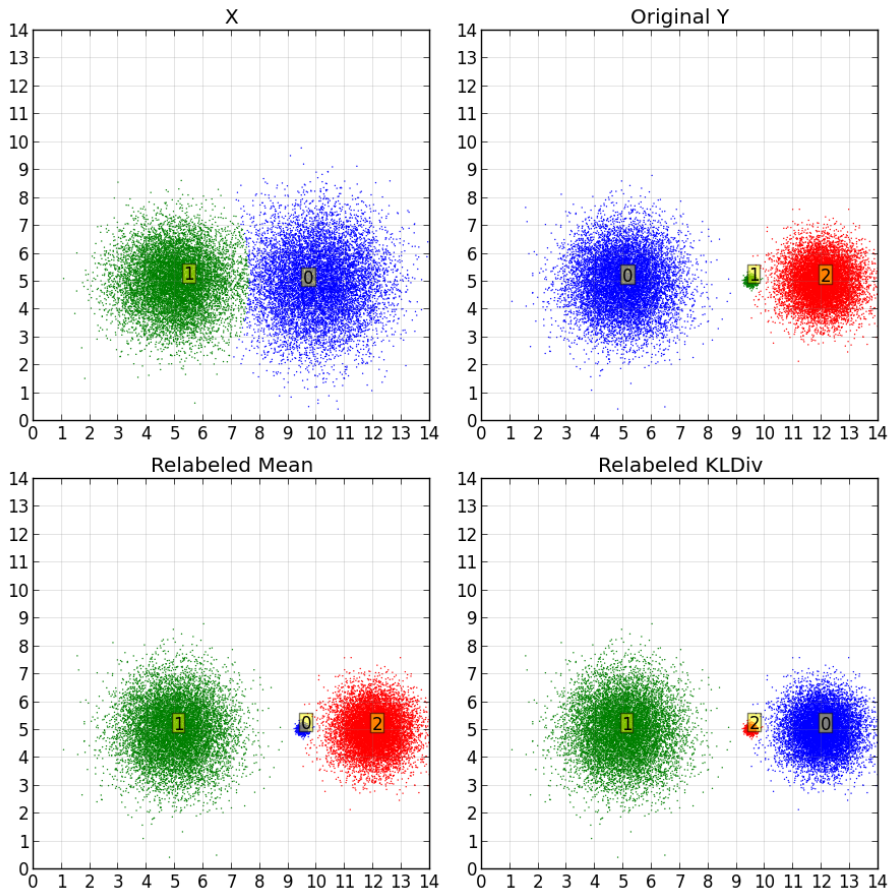

FIGURE 4.9: Illustration of problems using Euclidean distance between means. Cluster 1 in the Y data set has the same mean as cluster 0 in X. Because of this using Euclidean distance relabels it as cluster 0. Using Kullback-Leibler divergence cluster 2 is found to be a better match and relabeled as 0.

labeled the cytokine positive CD4 and CD8 populations, labeled as mode 10, and 18 respectively, and was chosen as the reference sample for relabeling. Relabeling was performed with each of the three dissimilarity measures, both with and without a maximum acceptable cost.

Scatter plots showing relabeling of sample E05 using mean distance is provided in the second row of Figure 4.10, and relabeling using mean distance using a max cost of 2 is provided in the third row of the same figure. The relative frequency of events associated with each mode before and after relabeling is shown in Figure 4.11.

Similarly, an example of relabeling using misclassification is provided in the fourth row of Figure 4.10, and relabeling using misclassification with a maximum cost of 0.2 is provided in the fifth row of the same figure. The relative frequency of number of events associated with each mode before and after relabeling is provided in Figure 4.12.

Finally, scatter plots of relabeling using Kullback-Leibler divergence without a maximum cost, and with a maximum cost of 10 is provided in the bottom two rows of Figure 4.10. The relative frequency events associating each mode before and after relabeling is provided in Figure 4.13. In each of the six cases, relabeling correctly identifies the target CD4 and CD8 cytokine positive populations in the sample E05.

4.4 Discussion

When models are fit individually, there is a need to assign matching labels to "matching" clusters for comparative analysis. Pooling all data and fitting with a single model is a simple way to generate a consensus model, but is not feasible for very large collections of data due to the increased computational time and system resources needed to fit the data. Pooling may also mask clusters unique to specific samples. Choosing a reference data set and using the model generated from that to label the other data sets is another alternative. This assumes the reference data set is a good fit for all data sets, and can run into problems for populations that exist in some samples but not in the reference. Hierarchical models, as those by Cron et al. (2013), eliminates the need for relabeling and generates consistent labels across all the sample in a way that avoids the limitations of pooling and reference models. However, hierarchical models still require all samples to be available, which may not be feasible in all situations

For relabeling to work, there must be some measure of dissimilarity between

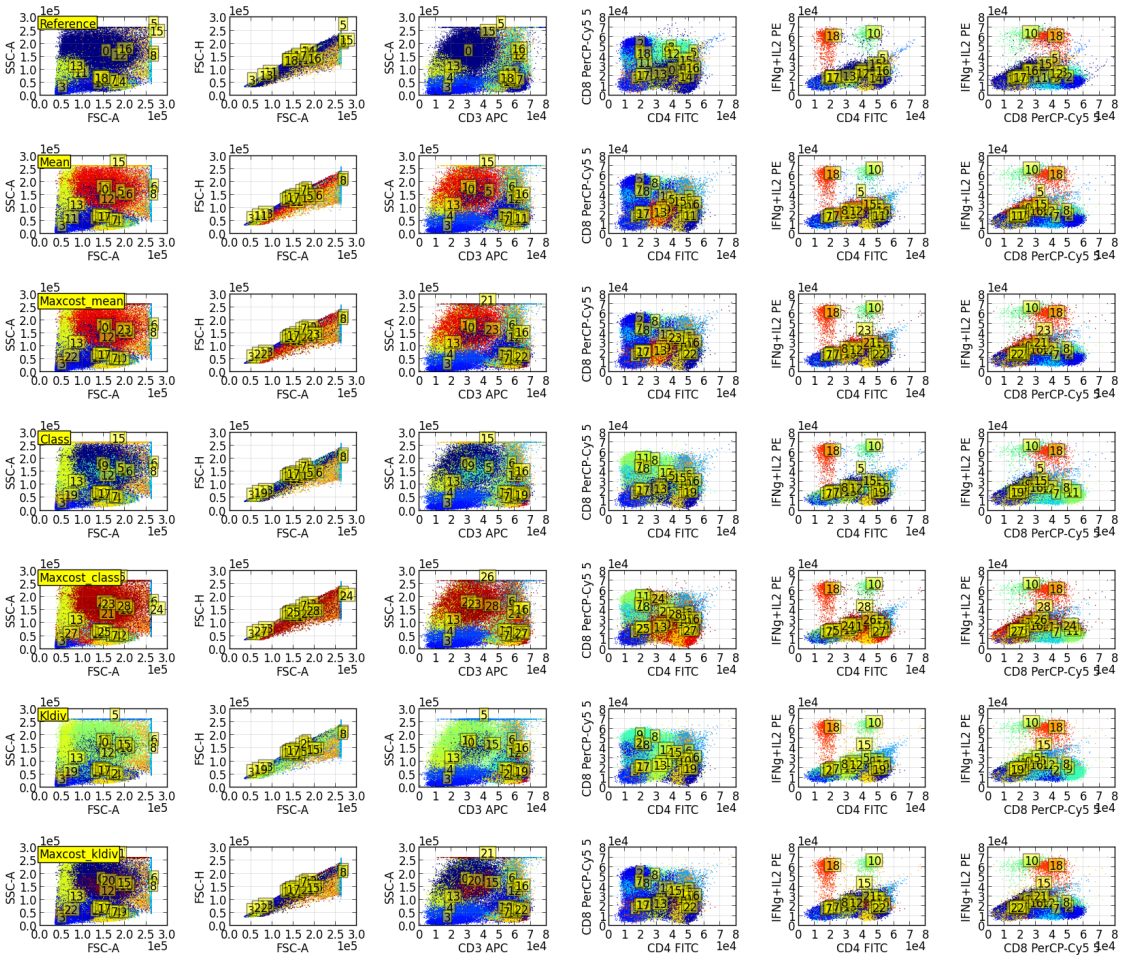

FIGURE 4.10: Relabeling of modes to a reference data set using all three dissimilarity measures, both with and without a max cost. The top row shows scatter plots of the reference sample E06. The second row shows the relabeling of sample E05 using Euclidean distance. The third row shows the same sample relabeled using Euclidean distance with a maximum cost of 2. The fourth row shows sample E05 using misclassification, and the fifth using misclassification with a maximum cost of 0.8. The bottom two rows shows scatter plots of sample E05 using relabeling using Kullback-Leibler divergence, without max cost on row six and with a maximum cost of 10 on the bottom row. Of note is the correct relabeling of the two cytokine positive modes, 16, and 10 across the samples.

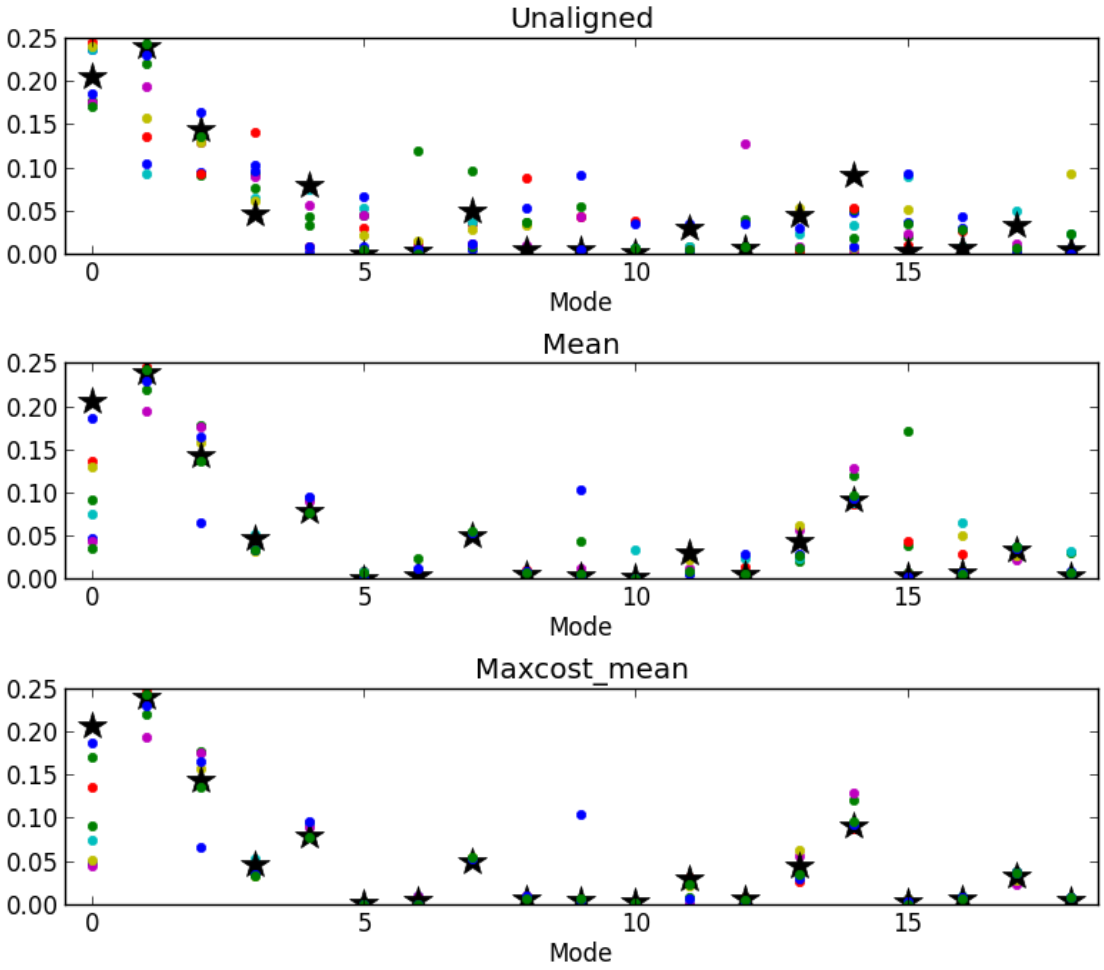

FIGURE 4.11: Frequency of events associated with each mode before and after relabeling relabeling using Euclidean distance between modes, both without and with a maximum cost of 2. Stars represent frequency of modes from data set G6904VJT-07_E06 used as reference in relabeling. The top plot shows the frequency of events associated with each mode before relabeling. The second plot shows the frequency of events associated with relabeled modes using Euclidean distance. The bottom plot shows the frequency of events associated with relabeled modes using Euclidean distance with a maximum cost of 2. The distribution of the frequency of many of the modes has a smaller variance after relabeling.

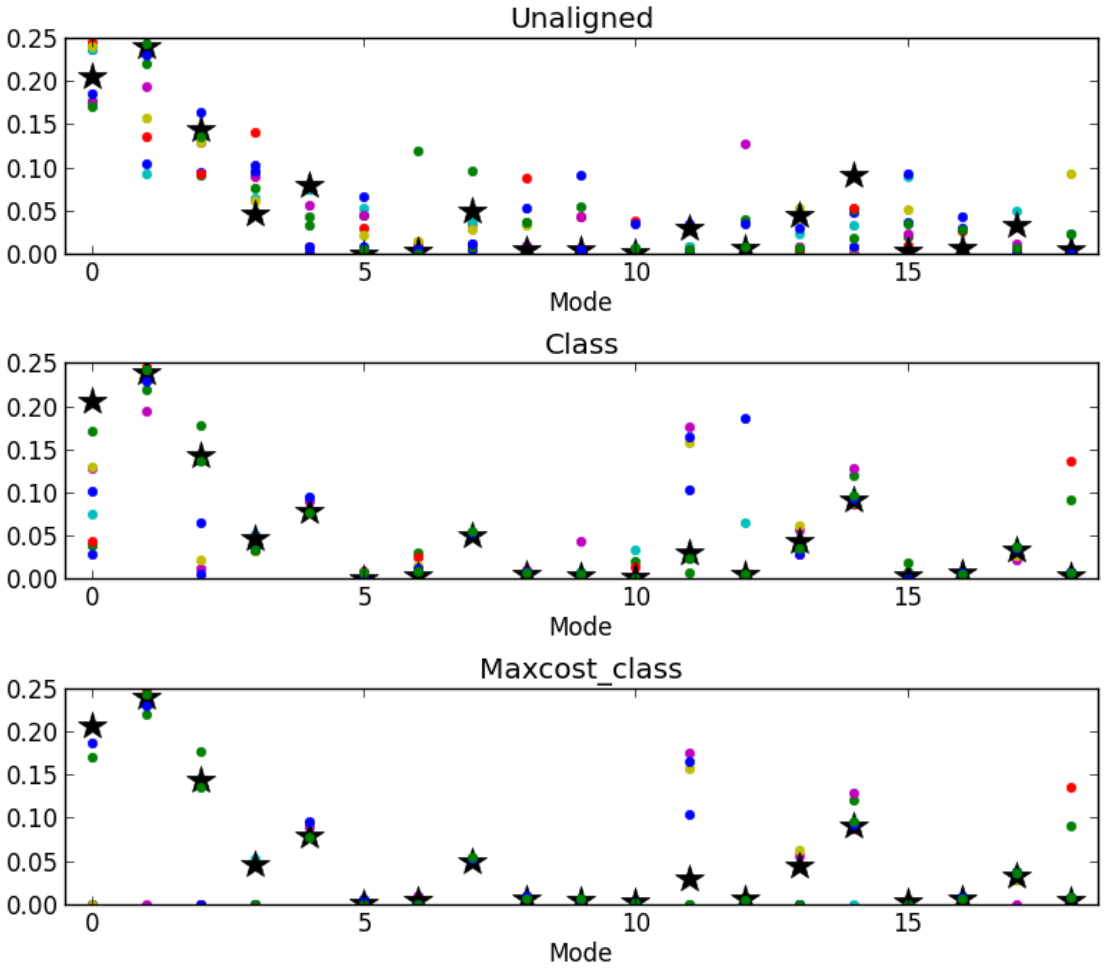

FIGURE 4.12: Frequency of events associated with each mode before and after relabeling relabeling using misclassification between modes, both without and with a maximum cost of 0.2. Stars represent frequency of modes from data set G6904VJT-07_E06 used as reference in relabeling. The top plot shows the frequency of events associated with each mode before relabeling. The second plot shows the frequency of events associated with relabeled modes using misclassification. The bottom plot shows the frequency of events associated with relabeled modes using misclassification with a maximum cost of 0.2. The distribution of the frequency of many of the modes has a smaller variance after relabeling.

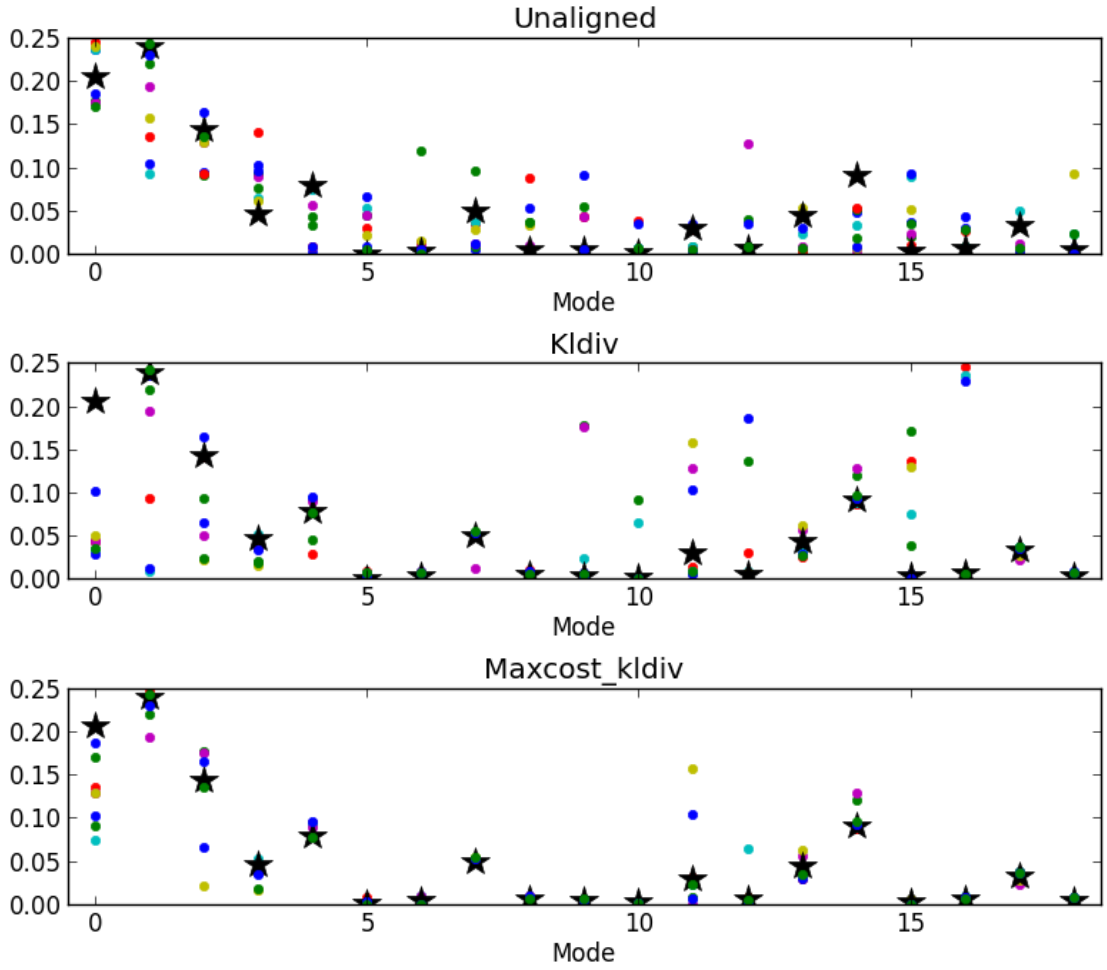

FIGURE 4.13: Frequency of events associated with each mode before and after relabeling relabeling using Kullback-Leibler divergence between modes, both without and with a maximum cost of 10. Stars represent frequency of modes from data set G6904VJT-07_E06 used as reference in relabeling. The top plot shows the frequency of events associated with each mode before relabeling. The second plot shows the frequency of events associated with relabeled modes using Kullback-Leibler divergence. The bottom plot shows the frequency of events associated with relabeled modes using Kullback-Leibler divergence with a maximum cost of 10. The distribution of the frequency of many of the modes has a smaller variance after relabeling.

clusters. Each of the three dissimilarity measures proposed has advantages and disadvantages. Euclidean distance is easy to calculate, but does not take into account the shape or size of the modes being relabeled. This can cause some small populations to be incorrectly relabeled.

Misclassification can run into problems identifying differing populations, as when classifying events, every event is necessarily assigned to some mode. A trivial example of this would be to consider two samples each consisting of a single mode. No mater how great the difference in sample distributions, the two modes would be matched.

Kullback-Leibler divergence incorporates not only the shape of the modes, but also position. The lack of a closed form of the Kullback-Leibler divergence for many distributions poses a problem if the mixture model used is not comprised of supported distributions. In this case, some approximation of the Kullback-Leibler divergence needs to be used, for example Monte Carlo estimates.

Determining the maximum cost of relabeling for max cost relabeling can be challenging. The acceptable maximum cost is dependent on which dissimilarity measure is employed, and the consistency of the data samples. Examining the distribution of all entries in the cost matrices generated for relabeling can indicate a range of acceptable values of max cost, but determining the maximum cost remains an open problem.

5

Scaling Up to Large Scale Comparative Analysis

5.1 Introduction

Clinical flow cytometry studies can have multiple data samples from a large number of subjects. As the panels used in these studies grow in complexity, the need for automated tools also grows, as manual analysis scales poorly. The previous chapters have developed several tools that have the potential to increase the accuracy and cross sample comparability of automated analysis. In this chapter, we will present a selection of software tools to build a pipeline implementing the three previously described methods of data alignment, interesting event sub-sampling, and cluster relabeling to automate analysis of EQAPOL ICS data sets. The EQAPOL data sets consist of of twelve laboratories each with twenty seven samples per panel. The total data set comprises over three hundred FCS files per panel, illustrating well the need for automated analysis tools, as manual analysis of this many files is tedious and costly.

5.2 Methods

To build an automated pipeline for the analysis of the data from the EQAPOL program, we developed *fcm*, an open source python library for working with flow cytometry data. The methods for data alignment, biased sub-sampling, and cluster relabeling in the previous chapters were implemented and are part of *fcm*.

5.2.1 fcm

The goal of *fcm* is to provide a general purpose library for working with flow cytometry data. *fcm* provides routines for loading, compensating, and manipulating flow cytometry data, along with routines for model based analysis using Gaussian mixture models. *fcm* provides a consistent API for exploratory data analysis using tools such as IPython, writing pipelines for batch data analysis, and development of end user graphical or web based software packages for flow cytometry.

As a library for working with flow cytometry data, *fcm* provides routines for loading data from FCS files, standard data storage files for flow cytometry data. *fcm* also provides methods to compensate data sets, and perform standard log or logicle transforms for visualization of flow data.

The central object representing flow data in *fcm* is the *FCMdata* object. *FCMdata* objects provide access to metadata stored in the FCS file, along with the numerical values for the scatter and fluorescent intensities of each event in a standard python *numpy.array*. By using *numpy.array*s for the storage of event data *FCMdata* objects can be passed to functions and methods expecting *numpy.array*s, and hence can tap into the large universe of python scientific software, including tools for numerical analysis and visualization.

fcm also provides tools for performing basic gating based analysis. *fcm* contains objects representing standard two dimensional polygonal and quadrant gates, along

with single dimensional threshold and interval gates, with a consistent API for applying gates to *FCMdata* objects. In addition to traditional gating, *fcm* provides *DropChannel* objects that remove whole channels from *FCMdata* objects, reducing the dimension of the data set.

In addition to traditional gating based analysis, *fcm* provides mechanisms for model based analysis using Gaussian mixture models (GMMs). *DPMixtureModel* objects manage GMM hyper-parameters and can be used to estimate the distribution of *FCMdata* or *numpy.array* objects using a Dirichlet process mixture model. Using a *DPMixtureModel* to fit a itFCMdata object results in a *DPMixture* object, representing the weights, means and co-variances of the estimated distribution. *DPMixture* objects then can be used to classify, calculate the likelihood or probability of arbitrary data sets. Additionally *fcm* provides a object to fit multiple *FCMdata* objects simultaneously using a Hierarchical Dirichlet process mixture model.

Finally *fcm* provides a set of convenience functions for working with *matplotlib*, a commonly used python plotting package. The *fcm.graphics* package provides utilities for generating density based pseudo-color plots commonly seen in flow cytometry, and for setting appropriate axis scaling and labels for the logicle or log transformed display.

5.2.2 Pipeline

The pipeline developed consists of the following stages

1. **Loading and preprocessing**

   Individual FCS files will be loaded, compensated and basic quality control to remove gross outliers where necessary.

2. **Generate reference sample**

   Interesting event subsampling of one of the samples from a single laboratory

will be used to generate a subsample enriched for the cytokine positive CD4 and CD8 cells of interest for each laboratory.

3. **Data alignment**

   The interesting event subsample for each laboratory will be aligned to the reference subsample generated in the previous step. The resulting data alignment parameters will be used to transform each data set in that lab.

4. **Clustering**

   Each sample, including the synthetic reference sample, will be clustered using Dirichlet process Gaussian mixture models

5. **Cluster relabeling**

   Clusters will be relabeled using the sample used to generate the subsample as a reference

6. **Target cell quantification**

   Frequencies of target populations of cytokine positive CD4 and CD8 cells will be calculated, along with various figures generated

5.3 Results

The four color EQAPOL data samples were run through the previously described pipeline. The data sets consist of 27 samples from 12 labs. The 27 samples from each lab consist of 9 samples from three subjects. The three subjects were common across all labs. Each of the samples from each subject included two stimulation samples and a brefeldin negative control sample. Each of these samples has two technical replicates. A total of 324 samples were processed. Subjects were labeled A,C, and E. Samples number 1,2 and 3 are the brefeldin negative control and technical replicates. Samples numbered 5,6,7 are a cytomegalovirus stimulation (CMV

pp65) peptide pool sample and technical replicates. Samples numbered 9,10,11 are a cytomegalovirus, Epstein-Barr virus and influenza virus (CEF) epitopic peptide stimulation and technical replicates.

Interesting event subsampling was performed on samples 6 from subject E, using sample 1 (brefeldin negative control) from subject E as a negative control, and sample 9 (CEF stimulation) from subject E as a positive control. The subsample size was set to be one fifth the total sample size, which corresponded to roughly 50,000 events for each lab.

For affine normalization a reference sample was chosen. The cross site Kullback-Leibler divergence was calculated between all sites. The reference site for affine normalization was chosen as the subsample with the lowest sum of Kullback-Leibler divergences against the other labs, as seen in Table 5.1. In this case the reference site was chosen as laboratory 31.

Table 5.1: Choosing a reference site. The subsamples for each lab was compared to all the other labs using Kullback-Leibler divergence between subsamples. The summed Kullback-Leibler divergence is displayed in the right column as "Sum of f_0". Lab labeled 031 was chosen as the reference due to having the lowest summed Kullback-Leibler divergence.

Lab	Sum of f_0
001	121.57408541
003	71.4792424607
004	76.8731717578
006	71.4475725884
007	74.8329361212
008	74.6719844847
010	149.461315853
011	68.6268845726
013	70.949087383
031	63.5807684587
044	89.7827256102
101	262.447737773

Subsamples were aligned to the subsample generated from lab 031. The trans-

forms generated by aligning each subsample were then applied to the 27 samples from the corresponding laboratory.

Clustering using Gaussian mixture models was then applied to each of the 324 samples. Clustering used prior means generated from the reference subsample. Events were then classified to posterior modes. Sample 6 from subject E from laboratory 031 was chosen as a reference for reclassification. Scatter plots for subject E sample 6 from laboratory 031 are shown in Figure 5.1. From this the target, CD3 positive, CD4 positive, cytokine positive populations were identified as modes 22, 28, and 36. The CD3 positive, CD8 positive, cytokine positive modes were identified as 8, 25, and 35. Scatter plots highlighting just these modes can be found in Figure 5.2

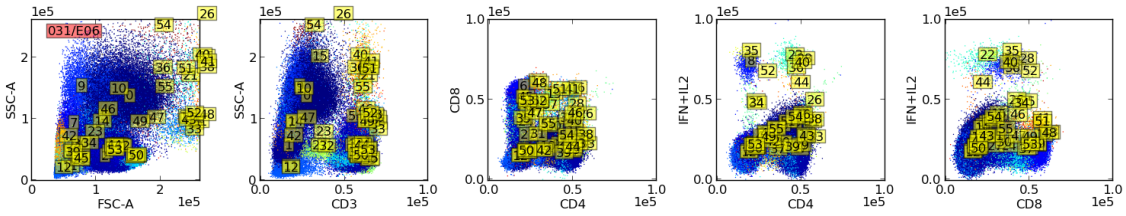

FIGURE 5.1: Scatter plots of sample 6 from subject E from laboratory 031. This sample was used as the reference sample for relabeling all other samples. Target cell populations of interest are represented by modes 8,25, and 35 for the cytokine positive CD8 T cell population, and 22, 28, and 36 for the cytokine positive CD4 T cell population.

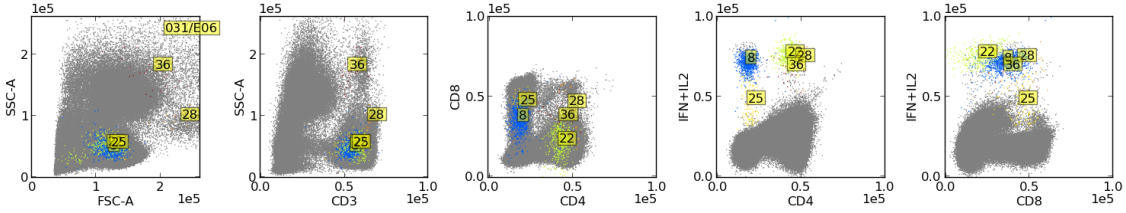

FIGURE 5.2: Scatter plots of sample 6 from subject E from laboratory 031 highlighting the target cell populations of interest. Target cell populations of interest are represented by modes 8,25, and 35 for the cytokine positive CD8 T cell population, and 22, 28, and 36 for the cytokine positive CD4 T cell population.

All samples were then relabeled using subject E sample 6 from laboratory 031 as a reference, using Kullback-Leibler divergence as the dissimilarity measure. The initial costs of realignment is shown in Figure 5.3. From this a maximum cost of 20 was chosen, to allow most relabels to occur yet prohibit very poor matches from being accepted.

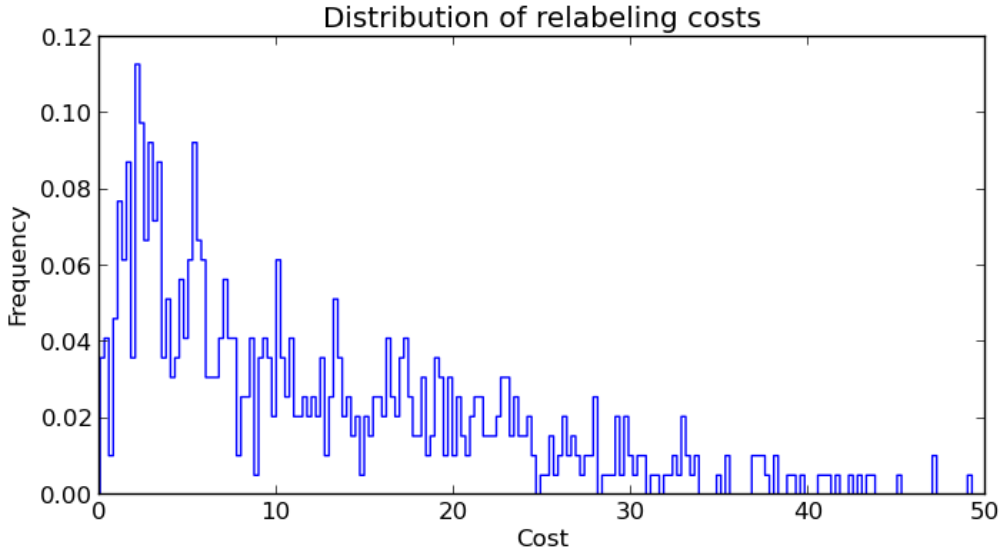

FIGURE 5.3: Distribution of values in the cost matrix for relabeling all populations using the Kullback-Leibler divergence to the model fit to subject E, sample 6, from laboratory 031 as a reference.

A plot of the target populations from lab 001, subject E , sample 10 can be seen in Figure 5.4. In this figure, the pipeline appears to have successfully identified the correct cytokine positive CD8 and CD4 T cell populations. The pipeline does miss populations. As seen in Figure 5.5, the bright cytokine CD8 positive population is missed.

The frequencies of identified cytokine positive CD4 and CD8 populations for subject A across all laboratories are shown in the top plot of Figure 5.6. Similarly, the frequencies for subject C are shown in the middle plot of Figure 5.6, and subject E in the bottom plot of the same figure.

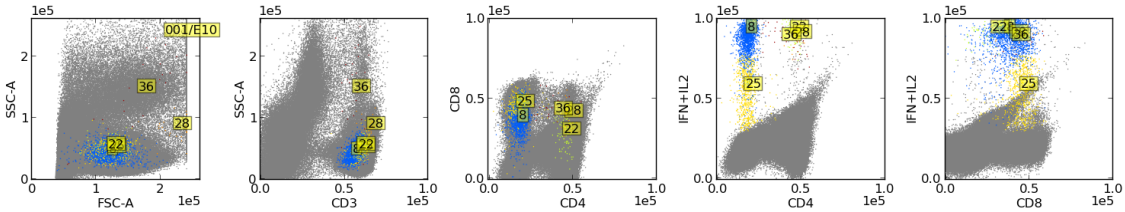

FIGURE 5.4: Scatter plots of sample 10 from subject E from laboratory 001 highlighting the target cell populations of interest. Target cell populations of interest are represented by modes 8, and 25, for the cytokine positive CD8 T cell population, and 22, 28, and 36 for the cytokine positive CD4 T cell population.

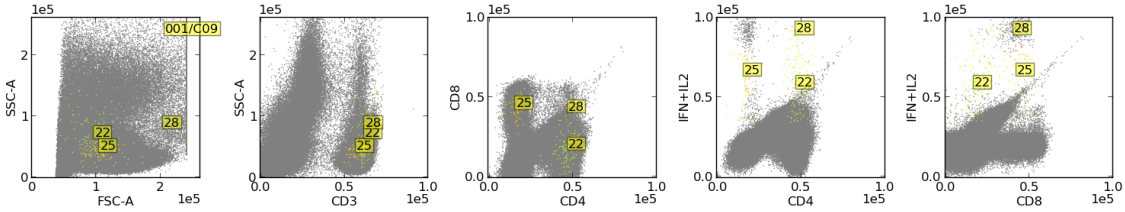

FIGURE 5.5: Scatter plots of sample 09 from subject C from laboratory 001 highlighting the target cell populations of interest. Target cell populations of interest are represented by mode 25, for the cytokine positive CD8 T cell population, and 22, and 28 for the cytokine positive CD4 T cell population. Relabeling failed to identify the bright cytokine positive CD8 positive population as being one of the desired modes.

5.4 Discussion

There is a need for automated analysis of flow cytometry data. Multi-center trials can produce such large numbers of samples that manual analysis would be a bottleneck. New technologies, such as mass-spectrometry based flow, or imaging cytometers can result in very large panels with 30 to 100 dimensions, capable of discriminating larger numbers of cell subset populations. Traditional gating based analysis scales poorly as panels become more complex, as it relies on two dimensional projections of the data.

Statistical mixture modeling provides an attractive method for automated analysis. However, mixture models can be computationally expensive, and there are interpretation challenges when using mixture models to compare cell subset frequen-

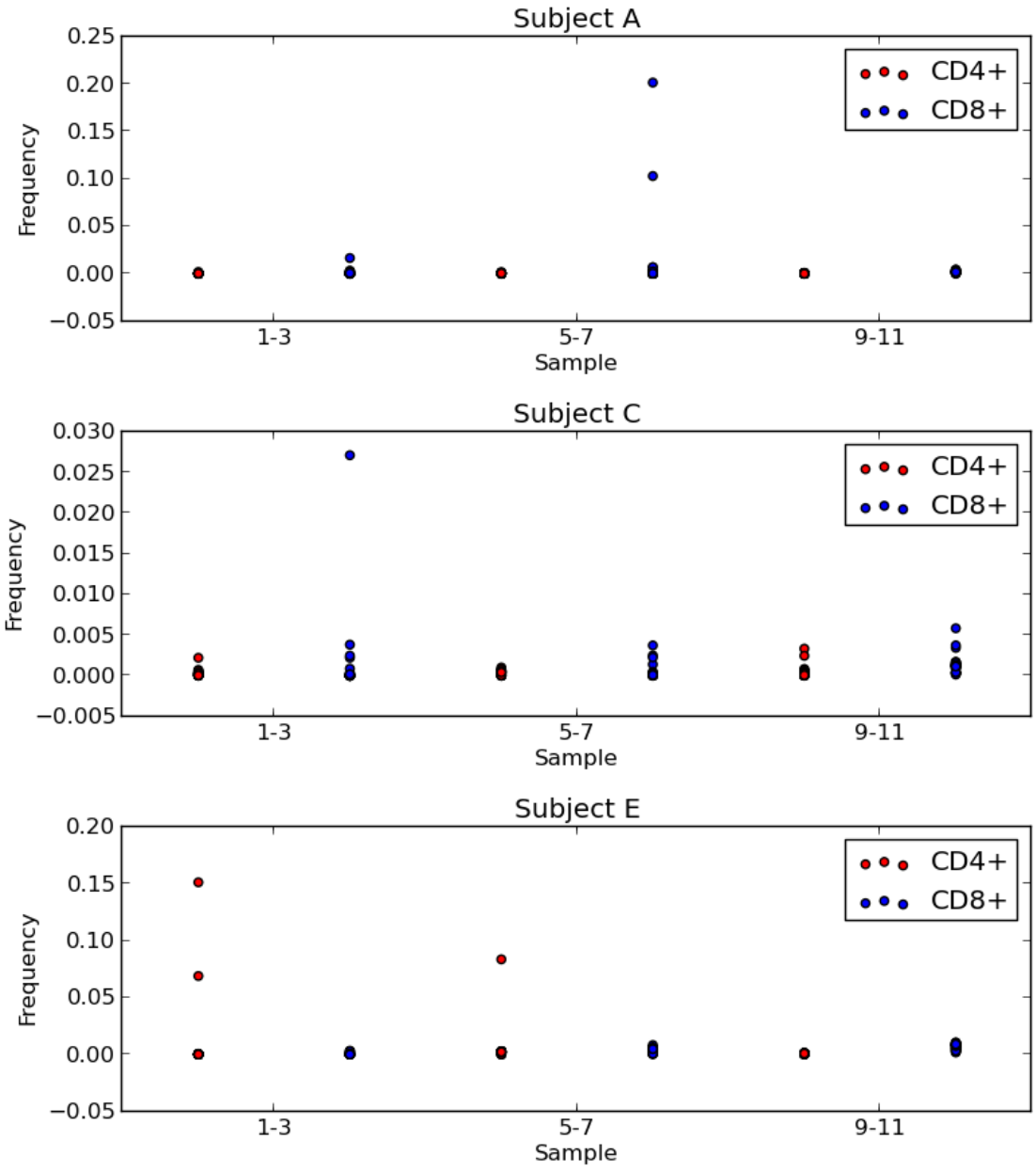

FIGURE 5.6: Frequency of cytokine positive CD4 and CD8 T cells from subject A in the top plot, subject C in the middle plot and subject E in the bottom plot as determined by automated analysis. Samples 1-3 are the brefeldin negative control. Samples 5-7 are the CMV PP65 group, and samples 9-11 are the CEF group.

cies across many samples. This thesis has described several methods to address these challenges.

fcm provides access to robust modeling methods along with the previously described novel alignment, subsampling and relabeling methods. It is currently being used for the analysis of proficiency data sets from EQAPOL, and the Association for Cancer Immunotherapy (CIMT). It is actively being developed and can provide a viable platform for development of new techniques or applications for flow cytometry The high level nature of *fcm* allows for rapid development of new pipelines, or easy modification of existing routines.

Bibliography

N. Aghaeepour, R. Nikolic, H. H. Hoos, and R. R. Brinkman. Rapid cell population identification in flow cytometry data. *Cytometry A*, 79(1):6–13, Jan 2011. doi: 10.1002/cyto.a.21007. URL http://dx.doi.org/10.1002/cyto.a.21007.

N. Aghaeepour, G. Finak, F. A. P. C. , D. R. E. A. M. C. , H. Hoos, T. R. Mosmann, R. Brinkman, R. Gottardo, and R. H. Scheuermann. Critical assessment of automated flow cytometry data analysis techniques. *Nat Methods*, 10(3):228–238, Mar 2013. doi: 10.1038/nmeth.2365. URL http://dx.doi.org/10.1038/nmeth.2365.

M. J. Boedigheimer and J. Ferbas. Mixture modeling approach to flow cytometry data. *Cytometry A*, 73(5):421–429, May 2008. doi: 10.1002/cyto.a.20553. URL http://dx.doi.org/10.1002/cyto.a.20553. mixture of Guassians via EM.

B. M. Bolstad, R. A. Irizarry, M. Astrand, and T. P. Speed. A comparison of normalization methods for high density oligonucleotide array data based on variance and bias. *Bioinformatics*, 19(2):185–193, Jan 2003.

F. Bourgeois and J.-C. Lassalle. An extension of the Munkres algorithm for the assignment problem to rectangular matrices. *Commun. ACM*, 14(12):802–804, December 1971. ISSN 0001-0782. doi: 10.1145/362919.362945. URL http://doi.acm.org/10.1145/362919.362945.

C. Chan, F. Feng, J. Ottinger, D. Foster, M. West, and T. B. Kepler. Statistical mixture modeling for cell subtype identification in flow cytometry. *Cytometry A*, 73(8):693–701, Aug 2008. doi: 10.1002/cyto.a.20583. URL http://dx.doi.org/10.1002/cyto.a.20583.

A. Cron, C. Gouttefangeas, J. Frelinger, L. Lin, S. K. Singh, C. M. Britten, M. J. P. Welters, S. H. van der Burg, M. West, and C. Chan. Hierarchical modeling for rare event detection and cell subset alignment across flow cytometry samples. *PLoS Comput Biol*, 9(7):e1003130, 07 2013. doi: 10.1371/journal.pcbi.1003130. URL http://dx.doi.org/10.1371%2Fjournal.pcbi.1003130.

A. J. Cron and M. West. Efficient classification-based relabeling in mixture models. *The American Statistician*, 65(1):16–20, 2011. doi: 10.1198/tast.2011.10170. URL http://www.tandfonline.com/doi/abs/10.1198/tast.2011.10170.

G. Finak, A. Bashashati, R. Brinkman, and R. Gottardo. Merging mixture components for cell population identification in flow cytometry. *Adv Bioinformatics*, page 247646, 2009. doi: 10.1155/2009/247646. URL http://dx.doi.org/10.1155/2009/247646.

F. Hahne, A. H. Khodabakhshi, A. Bashashati, C.-J. Wong, R. D. Gascoyne, A. P. Weng, V. Seyfert-Margolis, K. Bourcier, A. Asare, T. Lumley, R. Gentleman, and R. R. Brinkman. Per-channel basis normalization methods for flow cytometry data. *Cytometry A*, 77(2):121–131, Feb 2010. doi: 10.1002/cyto.a.20823. URL http://dx.doi.org/10.1002/cyto.a.20823.

J. Hershey and P. Olsen. Approximating the Kullback Leibler divergence between gaussian mixture models. In *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, volume 4, pages IV–317–IV–320, 2007. doi: 10.1109/ICASSP.2007.366913.

L. Lin, C. Chan, S. R. Hadrup, T. M. Froesig, Q. Wang, and M. West. Hierarchical Bayesian mixture modelling for antigen-specific T-cell subtyping in combinatorially encoded flow cytometry studies. *Stat Appl Genet Mol Biol*, 12(3):309–331, Jun 2013. doi: 10.1515/sagmb-2012-0001. URL http://dx.doi.org/10.1515/sagmb-2012-0001.

K. Lo, R. R. Brinkman, and R. Gottardo. Automated gating of flow cytometry data via robust model-based clustering. *Cytometry A*, 73(4):321–332, Apr 2008. doi: 10.1002/cyto.a.20531. URL http://dx.doi.org/10.1002/cyto.a.20531.

R. F. Murphy. Automated identification of subpopulations in flow cytometric list mode data using cluster analysis. *Cytometry*, 6(4):302–309, Jul 1985. doi: 10.1002/cyto.990060405. URL http://dx.doi.org/10.1002/cyto.990060405.

S. Pyne, X. Hu, K. Wang, E. Rossin, T.-I. Lin, L. M. Maier, C. Baecher-Allan, G. J. McLachlan, P. Tamayo, D. A. Hafler, P. L. De Jager, and J. P. Mesirov. Automated high-dimensional flow cytometric data analysis. *Proceedings of the National Academy of Sciences*, 106(21):8519–8524, 2009. doi: 10.1073/pnas.0903028106. URL http://www.pnas.org/content/106/21/8519.abstract. FLAME.

Y. Qian, C. Wei, F. Eun-Hyung Lee, J. Campbell, J. Halliley, J. A. Lee, J. Cai, Y. M. Kong, E. Sadat, E. Thomson, P. Dunn, A. C. Seegmiller, N. J. Karandikar, C. M. Tipton, T. Mosmann, I. Sanz, and R. H. Scheuermann. Elucidation of seventeen human peripheral blood B-cell subsets and quantification of the tetanus response using a density-based method for the automated identification of cell populations in multidimensional flow cytometry data. *Cytometry B Clin Cytom*, 78 Suppl 1: S69–S82, 2010. doi: 10.1002/cyto.b.20554. URL http://dx.doi.org/10.1002/cyto.b.20554.

P. Qiu, E. F. Simonds, S. C. Bendall, K. D. Gibbs, Jr, R. V. Bruggner, M. D. Linderman, K. Sachs, G. P. Nolan, and S. K. Plevritis. Extracting a cellular hierarchy from high-dimensional cytometry data with SPADE. *Nat Biotechnol*, 29 (10):886–891, Oct 2011. doi: 10.1038/nbt.1991. URL http://dx.doi.org/10.1038/nbt.1991.

J. Quinn, P. W. Fisher, R. J. Capocasale, R. Achuthanandam, M. Kam, P. J. Bugelski, and L. Hrebien. A statistical pattern recognition approach for determining cellular viability and lineage phenotype in cultured cells and murine bone Marrow. *Cytometry A*, 71(8):612–624, Aug 2007. doi: 10.1002/cyto.a.20416. URL http://dx.doi.org/10.1002/cyto.a.20416.

N. Z. Shor and N. Zhurbenko. The minimization method using space dilatation in direction of difference of two sequential gradients. *Kibernetika*, 3:51–59, 1971.

S. K. Singh, B. Tummers, T. N. Schumacher, R. Gomez, K. L. M. C. Franken, E. M. Verdegaal, K. Laske, C. Gouttefangeas, C. Ottensmeier, M. J. P. Welters, C. M. Britten, and S. H. van der Burg. The development of standard samples with a defined number of antigen-specific T cells to harmonize T cell assays: A proof-of-principle study. *Cancer Immunol Immunother*, 62(3):489–501, Mar 2013. doi: 10.1007/s00262-012-1351-0. URL http://dx.doi.org/10.1007/s00262-012-1351-0.

J. Spidlen, A. Barsky, K. Breuer, P. Carr, M.-D. Nazaire, B. A. Hill, Y. Qian, T. Liefeld, M. Reich, J. P. Mesirov, P. Wilkinson, R. H. Scheuermann, R.-P. Sekaly, and R. R. Brinkman. GenePattern flow cytometry suite. *Source Code Biol Med*, 8(1):14, 2013. doi: 10.1186/1751-0473-8-14. URL http://dx.doi.org/10.1186/1751-0473-8-14.

M. A. Suchard, Q. Wang, C. Chan, J. Frelinger, A. Cron, and M. West. Understanding GPU programming for statistical computation: Studies in massively parallel massive mixtures. *Journal of Computational and Graphical Statistics*, 19(2):419–438, 2010. doi: 10.1198/jcgs.2010.10016. URL http://amstat.tandfonline.com/doi/abs/10.1198/jcgs.2010.10016.

I. P. Sugr and S. C. Sealfon. Misty mountain clustering: Application to fast unsupervised flow cytometry gating. *BMC Bioinformatics*, 11:502, 2010. doi: 10.1186/1471-2105-11-502. URL http://dx.doi.org/10.1186/1471-2105-11-502.

H. Zare, P. Shooshtari, A. Gupta, and R. R. Brinkman. Data reduction for spectral clustering to analyze high throughput flow cytometry data. *BMC Bioinformatics*, 11:403, 2010. doi: 10.1186/1471-2105-11-403. URL http://dx.doi.org/10.1186/1471-2105-11-403.

Q. T. Zeng, J. P. Pratt, J. Pak, D. Ravnic, H. Huss, and S. J. Mentzer. Feature-guided clustering of multi-dimensional flow cytometry datasets. *J Biomed Inform*,

40(3):325–331, Jun 2007. doi: 10.1016/j.jbi.2006.06.005. URL http://dx.doi.org/10.1016/j.jbi.2006.06.005.

Biography

Jacob Frelinger was born in 1978, in Santa Monica, California. He spent his childhood in Chapel Hill, and currently resides in Raleigh.